ETSI EN 301 931-2 V1.1.2 (2001-09)

European Standard (Telecommunications series)

Intelligent Network (IN);
Intelligent Network Capability Set 3 (CS3);
Intelligent Network Application Protocol (INAP);
Protocol specification;
Part 2: SCF-SSF interface

ETSI

Reference

DEN/SPAN-03063/1-2

Keywords

CS3, CTM, IN, INAP, protocol, UPT

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00   Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or
perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF).
In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive
within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status.
Information on the current status of this and other ETSI documents is available at http://www.etsi.org/tb/status/

If you find errors in the present document, send your comment to:
editor@etsi.fr

Copyright Notification

ETSI

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://www.etsi.org/ipr).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This European Standard (Telecommunications series) has been produced by ETSI Technical Committee Services and Protocols for Advanced Networks (SPAN).

The present document is part 2 of a multi-part deliverable covering Intelligent Network (IN); Intelligent Network Application Protocol (INAP); Capability Set 3 (CS3); Protocol specification, as identified below:

Part 1: "Common aspects";

Part 2: "SCF-SSF interface";

Part 3: "SCF-SRF interface";

Part 4: "SDLs for SCF-SSF interface".

The present document and parts 1, 3 and 4 define the Intelligent Network (IN) Application Protocol (INAP) for IN Capability Set 3 (IN CS-3). The present document and parts 1, 3 and 4 define the INAP for IN CS-3 based upon ETSI Core INAP CS-2 (EN 301 140-1) and ITU-T IN CS3 Recommendation Q.1238 (1999).

The structure of the present document and parts 1, 3 and 4 follows the ITU-T Recommendation Q.1238 rather than that usual for an ETSI deliverable.

National transposition dates	
Date of adoption of this EN:	31 August 2001
Date of latest announcement of this EN (doa):	30 November 2001
Date of latest publication of new National Standard or endorsement of this EN (dop/e):	31 May 2002
Date of withdrawal of any conflicting National Standard (dow):	31 May 2002

1   Scope

The present document specifies the protocol on the SSF-SCF interface and provides a description of the aspects of the SSF and SCF Functional Entities, which are involved in the realization of this interface.

2   References

All documents referred to in the present document are identified in EN 301 931-1.

3   Definitions and abbreviations

3.1   Definitions

For the purposes of the present document, the terms and definitions given in EN 301 931-1 apply.

3.2   Abbreviations

For the purposes of the present document, the abbreviations and acronyms given in EN 301 931-1 apply.

4   Introduction

As a complement to the CCF/SSF parts described herein in the main text, the Specification and Description Language (SDL) has been used in the definition of modelling the SSF - SCF protocol behaviour for Call Party Handling (CPH). These SDLs have been substantially validated to ensure that the Connection View model can be implemented.
The SDL part is described in EN 301 931-4.

5   Relationships

5.1   SSF-CCF relationship

Call/service processing builds upon the current call process infrastructure of existing digital exchanges. It does so by using a generic model of existing call control functionality to process basic two-party calls, then adding service switching functionality to invoke and manage IN service logic. Once invoked, IN service logic is executed under the control of service control functionality, in conjunction with service data functionality. With this distributed approach to call/service processing, the existing call control functionality retains ultimate responsibility for the integrity of calls, as well as for the control of call processing resources.

The following call/service processing assumptions apply:

(1)   Call control and service switching functionality are tightly coupled; thus the relationship between SSF and CCF is not standardized.

(2)   A call is either between two or more end users that are external to the network and addressable via a directory number or combination of directory number and bearer capability, or a call is between one or more end users and the network itself.

(3)   A call may be initiated by an end user, or by an SCF within the network on behalf of an end user. To supplement a call, IN service logic may either be invoked by an end user served by an IN exchange, or by the network on behalf of an end user.

(4) A call may span multiple exchanges. As such, each exchange only controls the portion of the call in that exchange - call processing is functionally separated between exchanges. IN service logic invoked on IN exchanges in such an inter-exchange call are managed independently by each IN exchange.

(5) Existing exchanges can be viewed as having two functionally separate sets of call processing logic that co-ordinate call processing activities to create and maintain a basic two-party call. This functional separation is provided between the originating portion of the call and the terminating portion of the call. This functional separation should be maintained in an IN exchange to allow IN service logic invoked on the originating portion of the call (i.e. on behalf of the calling party) to be managed independently of IN service logic invoked on the terminating portion of the call (i.e. on behalf of the called party).

(6) It is desirable to allow multiple IN-supported service logic instances to be simultaneously active for a given end user. It is also recognized that non-IN service logic will continue to exist in the network. As such, service feature logic instances mechanisms should:

　i) determine which service logic to invoke for a given service request.
This mechanism should select the appropriate IN-supported service logic or non-IN-supported service logic, and block the invocation of any other service logic for that particular service request;

　ii) limit simultaneously active IN- and non-IN-supported service logic instances;

　iii) ensure that simultaneously active IN-supported service logic instances adhere either to the single-ended, single point of control (SPC) restriction or to the single ended, multiple points of control (MPC) capability as introduced with IN CS-3. For each half call a CS-3 compliant CCF/SSF may behave either according to SPC rules or MPC rules depending on trigger table information provided at IN service triggering.

(7) The distributed approach and added complexity of call/service processing requires mechanisms for fault detection and recovery, allowing graceful termination of calls and appropriate treatments for end users.

5.2　SSF-SCF relationship

The SCF-SSF relationship is used for communication between an SCF and an SSF in the public network. This relationship, with the aid of possible other relationships (e.g. the SCF-SRF, SCF-SCF and SCF-SDF), provides a variety of services and service features.

Details of service drivers for IN CS-3 can be found in ITU-T Recommendation Q.1231.

A relationship between the SCF and SSF is established either as a result of the SSF sending a request for instruction to the SCF, or at the request of the SCF for initiation of a call or for some non call-related reason.

A relationship between a SCF and a SSF is normally terminated at the request of the SCF. The SSF may also terminate the relationship, e.g. when no pending monitor requests prevails or in error cases.

A single SCF instance may have concurrent relationships with multiple SSF instances. A single SSF instance may have concurrent relationships with multiple SCF instances in case where the IN CS-3 Multiple Points of Control applies to a half call instance. If Single Point of Control applies a single SSF instance may only have a relationship with one SCF instance at a time for any given half-call instance. It should be noted that the selection of an SRF is not always performed by the SCF. In some cases selection is performed by an SSF, for example, where assist/hand-off procedures are being used. For details regarding the relationship to the SRF see EN 301 931-3.

6　CCF/SSF Model

An overview of the CCF/SSF FE model and its components as introduced in EN 301 931-1 is provided below.

The description is based upon figure 1 - "Separated single-ended service logic program instance related to a calling or called parties".

Functional separation
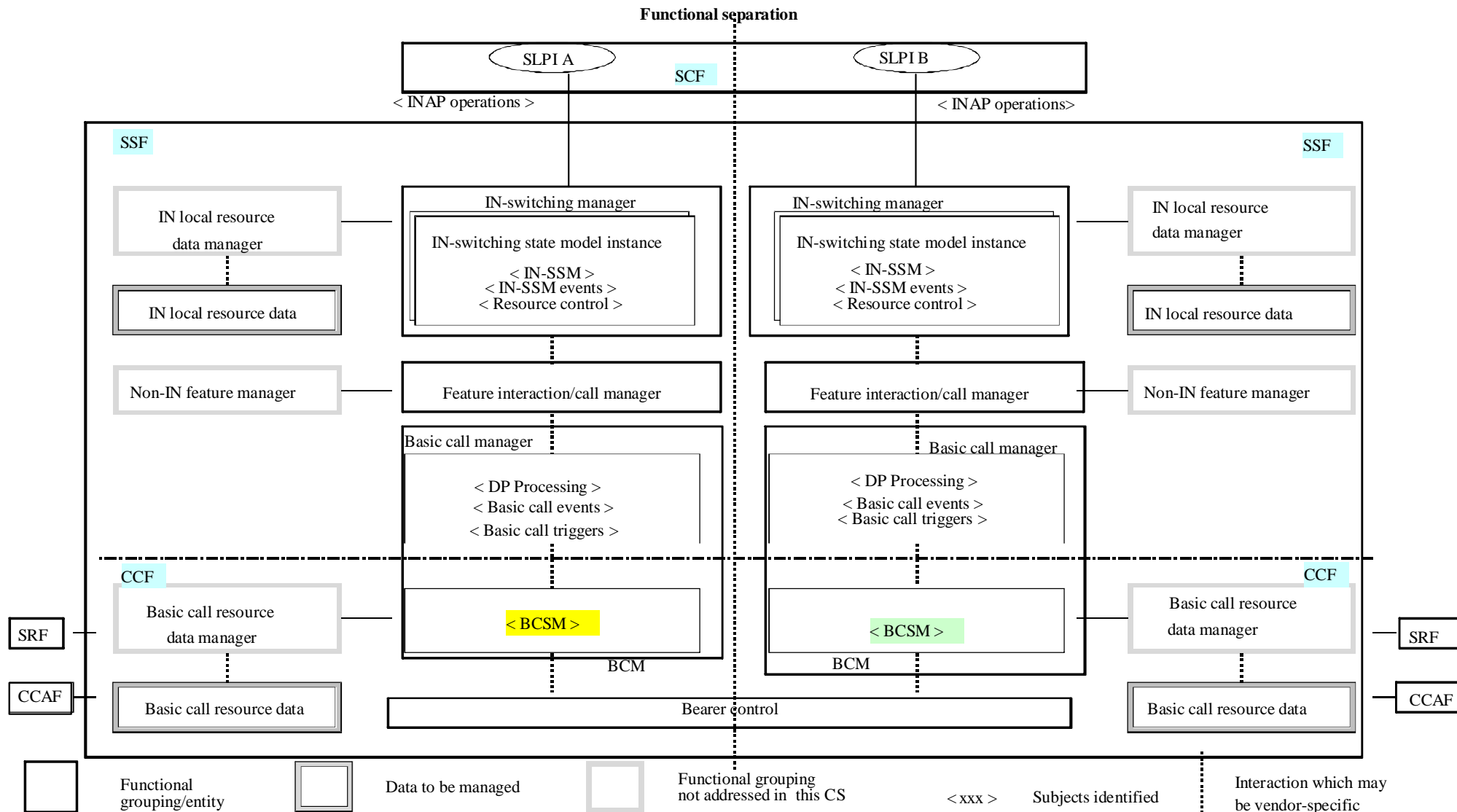
Figure 1: CCF/SSF model - Separated single-ended SLPIs related to calling and called parties

The purpose of this functional model is to provide a framework for call modelling subjects with respect to the CCF/SSF.

The functional separation in the CCF/SSFF model of call processes and resources serves to isolate single-ended service logic instances related to the calling party from single-ended service logic instances related to the called party for the same call, i.e. the "half call" model approach.

6.1   CCF/SSF Functional Model Components

The aspects of the CCF/SSF functional model described herein include the following components:

- BCM, the basic call manager;

- FIM/CM, the feature interactions manager/call manager;

- IN-SM, the IN-switching manager.

Furthermore a description is given of the following relationships between components:

- the BCM to the IN-SM;

- the BCM to the FIM/CM;

- the IN-SM to the FIM/CM;

- the FIM/CM to the non-IN FM.

Figure 2 gives a simplified view of the CCF/SSF model components and the message flows involved.
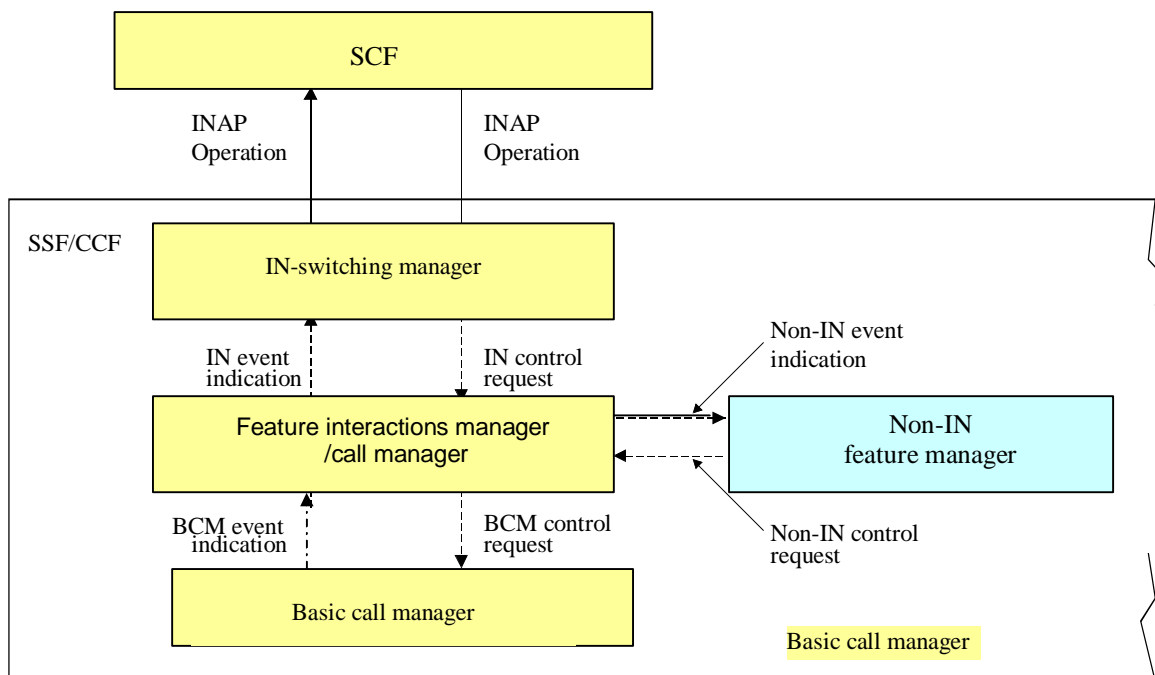
Figure 2: CCF/SSF Model for components and relationships

6.1.1   Basic Call Manager (BCM)

The Basic Call Manager, BCM provides an abstraction of a part of a switch that implements basic call and connection control to establish communication paths for users and to interconnect such communication paths. It detects basic call and connection control events that can lead to the invocation of IN service logic instances or should be reported to active IN service logic instances, and manages CCF/SSF resources required to support basic call and connection control.

The BCM also implements the O/T_BCSM and the DP processing.

The DP processing is the entity of the BCM that interacts with the FIM/CM.

When the SSF receives call-related parameters from the SCF, the BCM substitutes these parameters when allowed for the corresponding call information, and retains all other call information. This applies to all call processing-related messages.

6.1.2   Feature Interaction Manager/Call Manager (FIM/CM)

The Feature Interaction Manager/Call Manager provides mechanisms to support multiple concurrent instances of IN service logic instances and non-IN service logic instances on a single call.
In particular, the FIM/CM can prevent multiple instances of IN and non-IN service logic instances from being invoked. The FIM/CM integrates these interaction mechanisms with the BCM and IN-SM to provide the SSF with a unified view of call/service processing internal to the SSF for a single call.

IN Service/Feature Interaction Management

The management of IN-IN service interaction should be done based on ServiceCompatibilityIDs, if ServiceCompatibilityIDs are supported in the network.
The management of IN service interaction with network-based features should be done based on ServiceInteractionIndicators. Global management of feature interaction also calls for conveying the ServiceCompatibilityIDs of all IN services invoked during a call and the ServiceInteractionIndicators via basic network signalling to the originating and to the terminating instances. Furthermore, the OCCRUI mechanism as enhanced in CS-3 allows for the conveying of information between service logic instances in the network also during service invocation e.g. for the purpose of controlling feature interactions.

The FIM/CM supports:

- Prevention of non-IN service logic instance invocation according to instructions from an IN service logic.

- Prevention of multiple IN service logic invocations according to ServiceCompatibilityIDs which may be assigned to an IN service by the basic network operator via administration.
  The FIM performs a generic compatibility check procedure using an administrative exclusion matrix for ServiceCompatibilityIDs.

- Allowing support for Multiple Points of Control (MPC) according to the defined DP Processing Rules for MPC in the BCM.

6.1.3   IN-Switching Manager (IN-SM)

The IN-Switching Manager interacts with the SCF in the course of providing IN service features to users. It provides the SCF with an observable view of CCF/SSF call/connection processing activities, and provides the SCF with access to CCF/SSF capabilities and resources. It also detects IN call/connection processing events that should be reported to active IN service logic instances, and manages SSF resources required to support IN service logic instances. The IN-SM interacts with the FIM/CM as described below.

Figure 2 shows a simplified view of the CCF/SSF and depicts only the important components and relationships. The major relationships are those between the SCF and the IN-SM (via the FEAM), between the IN-SM and the FIM/CM, and between the FIM/CM and the BCM. Of all the relationships described below, only the relationship between the SCF and IN-SM is external to the CCF/SSF, and is therefore subject to standardization. The other relationships are assumed to exist in order to understand and describe the CCF/SSF model.

6.1.4   CCF/SSF Model components relationships

In figure 2 all the relationships are identified in terms of types of events between components. An overview is provided below.

6.1.4.1   Relationship between SCF and IN-SM

SSF operation:

-   information from the IN-SM to the SCF (via the FEAM in the SSF) that reports a call/connection processing event, as well as the current state of the call/connection instance in which the event is detected.

SCF operation:

-   information from the SCF to the IN-SM (via the FEAM in the SSF) that requests the manipulation of the state of a call/service instance.

6.1.4.2   Relationship between IN-SM and FIM/CM

IN event indication:

-   information from the FIM/CM to the IN-SM that reports a call processing event, the current state of the call in which the event is detected, and whether the event is to be handled by a new instance of IN service logic or an existing active instance.

IN control request:

-   information from the IN-SM to the FIM/CM that indicates call/service processing functions requested by the SCF.

6.1.4.3   Relationship between FIM/CM and BCM

BCM event indication:

-   information from the BCM to the FIM/CM that reports a BCSM event and the current state of the BCSM in which the event is detected.

BCM control request:

-   information from the FIM/CM to the BCM that requests the manipulation of one or more BCSMs to influence call/service processing.

6.1.4.4   Relationship between FIM/CM and non-IN FM

The relationship that encompasses the interaction between IN based services and non-IN (i.e. switch based) services. An example of this kind of interworking between IN and non-IN switch based service is where the INAP Connect operation may result in an abnormal termination of IN call as the involved BCSM instance is deleted due to the invocation of a switch based call redirecting service.

Non-IN event indication:

-   information from the FIM/CM to the non-IN FM.

Non-IN control request:

-   information from the non-IN FM to the FIM/CM.

6.1.5 Typical Sequence of Model Actions

This clause describes a typical sequence of actions in the CCF/SSF model to illustrate the roles and relationships of the major model components. This illustration is not intended to imply or reflect any specific implementation. This scenario provides an example in which a new instance of an IN-SSM is invoked to provide an IN service feature to a user.

1) A user is interacting with the CCF/SSF via the signalling interface arrangement to request the setup of a call. The BCM creates a BCSM to represent the basic call control functions required to establish and maintain this call for the user.

2) In the course of call setup for the user, an event is detected in the BCSM associated with the user's call. BCSM processing is halted at the DP.

3) The BCM processes the event at a DP in the BCSM to determine if the event should be reported (i.e. it determines if the DP is armed and DP criteria are met). If so, it sends a BCSM event indication reporting the event to the FIM/CM, along with the state of the BCSM at the time the event was detected. If the BCM needs instructions on how to proceed, BCSM processing remains halted at the DP until instructions are received. If not, the BCM continues normal BCSM processing. Thus, two scenarios are possible:

   - the BCM determines that the event should not be reported; BCSM processing continues (e.g. no TDP armed);

   - the BCM determines that the event should be reported, and needs further instructions (e.g. TDP-R armed); BCSM processing is halted and the BCM may continue to detect additional events before receiving instructions (the handling of these additional events is not addressed in this example).

4) The FIM/CM receives and processes the BCM event indication to determine if the event is to be processed by an IN service logic instance or a non-IN service logic instance. It also determines if the event is to be processed by a new instance of a service logic instance or an existing active instance.

5a) If the BCM event is to be processed by a new instance of IN service logic, the FIM/CM sends an IN event indication to the IN-SM, which reported the event. The event indication provides the state of the BCSM in which the event was detected and indicates that a new instance of IN service logic is to be invoked. Go to step 6.

5b) If the BCM event is to be processed by a new instance of non-IN service logic, the FIM/CM sends a non-IN event indication to the non-IN FM that reported the event. The event indication provides the state of the BCSM in which it was detected and indicating that a new instance of non-IN service logic is to be invoked. The non-IN FM receives and processes the non-IN event, and invokes the appropriate non-IN service logic instance. The non-IN FM executes the non-IN service logic instance, sending non-IN control requests to the FIM/CM as necessary to realize the service feature (the handling of subsequent messages for such a non-IN service logic instance, if any, are not addressed in this example.

6) The IN-SM receives and processes the IN event indication. Since a new instance of an IN service logic instance is to be invoked, the IN-SM creates a new instance of an IN-SSM to represent the state of the user's call and connection in a manner accessible to service logic processing programs (SLPs) in the SCF (e.g. in connection points). SSF then sends an INAP operation (via the FEAM) to the SCF providing a view of the current state of the IN-SSM.

7) The SCF receives and processes the INAP operation from the SSF. Given that a new instance of IN service logic is to be invoked, the SCF invokes an SLP instance (SLPI) that realizes the desired service feature. The SLPI is provided a view of the current state of the IN-SSM. The SCF issues an INAP operation to the SSF to request the IN-FM to manipulate the state of the IN-SSM, and/or to perform the call associated out channel interaction as appropriate to realize the service feature. An INAP operation from SCF may indicate the set of events that should be reported to the SLPI (i.e. it indicates the set of BCSM and IN-SSM EDPs or/and the out-channel interaction related events (e.g. a reception of USI from the user) to be armed for this particular service logic instance).

8) The IN-SM receives the INAP operations from the SCF (via the FEAM) and processes the INAP operations to manipulate the state of the IN-SSM, and/or to perform the call associated out-channel interaction as requested. In doing so, it generates an IN control request to the FIM/CM. It also monitors the IN-SSM for the IN-SSM events indicated in the request (if any).

9) The FIM/CM receives and processes the IN control request, and determines if it is valid based on other active service logic instances. It then sends a BCM control request to the BCM to notify it of the functions to be performed and of any BCSM events or/and the our channel interaction events to monitor for.

10) The BCM receives and processes the BCM control request, and manipulates one or more BCSMs to satisfy the request. In manipulating the BCSMs, it performs the appropriate bearer control and resource control functions or/and the out-channel interaction. The BCM also monitors the BCSMs for the BCSM events or/and the out-channel interaction events indicated in the BCM control request (if any).

11) If the BCM detects a BCSM event in a BCSM, it repeats step 3 to send a BCSM event indication to the FIM/CM.

   If the out-channel interactions related event is detected by the BCM, the BCM sends the event indication to the FIM/CM.

12) The FIM/CM repeats step 4 to determine how to process the event. In this case, the event is for an active IN service logic instance. It sends an IN event indication to the IN-SM, indicating that the event is for an existing instance of IN service logic.

13) The IN-SM receives and processes the IN event indication as in step 6, with the following differences. Given that the event is for an existing instance of IN service logic, as represented by an existing IN-SSM instance. It updates the state of the existing IN-SSM to reflect the state of the user's connection(s) and/or request via the out-channel interaction, and reports the event and current IN-SSM state to the SCF in an INAP operation from the SSF. No new IN-SSM instance is created.

14) The SCF receives and processes the INAP operations from the SSF as in step 7, with the following difference. Given that the event is for an existing instance of IN service logic, as supported by an existing SLPI. It does not invoke a new instance of an SLP. The SLPI then repeats its actions in step 7 to send INAP operation(s) from the SCF to the SSF to request the IN-SM to manipulate the state of the IN-SSM or to perform the out-channel interaction, and to indicate the next set of EDPs or/and the out-channel interaction related events of interest, if any.

15) Steps 8-14 are repeated until the IN service logic instance is ended. The IN service logic instance ends when the SLPI is no longer interested in any EDPs, or CCF/SSF processing has progressed beyond the point at which any EDPs can be encountered.

Details for the different components are provided in subsequent clauses.

6.2 Call Control Function

The Call Control Function (CCF) provides an abstraction of the part of a switch which implements basic call and connection control to establish communication paths for users, and to interconnect such communication paths.
It comprises the following types of objects:

- a Basic Call Controller;

- a Bearer Controller;

- Signalling Terminations;

- Bearer Terminations.

   NOTE: The Charging Processes (REG, DET, and GEN) could also be explicitly identified as an additional set of objects.

The Core INAP CS3 model consists of two half-calls, an originating (SSF_CCF_A) and a terminating (SSF_CCF_B) one.

The BCSM is supposed to model existing switch processing of a basic two-party call and reflects the functional separation between the originating and terminating portions of calls.

In this way the full functionality of the inter-working between the O_BCSM and the T_BCSM is catered for.

Since the BCSM is generic, it may describe events that do not apply to certain access arrangements (e.g. analogue signalling systems).

The Core INAP CS3 Model supports different Abstract Signalling Primitive interface types:

- the Signalling Termination Abstract Signalling Primitive interface;

- the Inter-BCSM Interface (IBI) between half calls;

- the internal interface to/from the SSF.

6.2.1 Basic Call Controller Process

6.2.1.1 Overview

The Basic Call Controller (BCC) is responsible for the overall co-ordination of basic call/connection processing in the CCF. It centres on the Basic Call State Model as defined in annex A of ITU-T Recommendation Q.1204. Since the BCSM is generic, it may describe events that do not apply to certain access arrangements (e.g. analogue signalling systems). This is taken into account in the Signalling Terminations.

The activities of the Basic Call Controller include:

- creating and deleting BCSM instances;

- routing signals (Abstract Signalling Primitives) between BCSM instances and Signalling Terminations;

- requesting services from the Bearer Controller (BRC).

The Basic Call Controller is the interface to the SSF. This relationship involves:

- executing instructions that request the manipulation of one or more BCSM instances;

- reporting events (i.e. DP) received from BCSM instances.

The Basic Call Controller is also in charge of relaying non-call processing related events between the SSF and the Signalling Terminations (e.g. USI information) or between the SSF and other local processes (e.g. charging information).

When executing instructions received from the SSF, the Basic Call Controller is responsible for the coherence between the configuration of the underlying bearer connections and the state of the Signalling Terminations FSM. As such, it may send an Abstract Signalling Primitive to a Signalling Termination in order to bring the FSM is the correct state.

EXAMPLE: As a result of an SCF instruction (e.g. MergeCallSegments), the Basic Call Controller Process may request the Bearer Controller Process to establish a physical connection between two parties, while the Signalling Termination associated to the originating half-call is not in the appropriate state (i.e. no answer received). In that case, the Basic Call Controller Process is responsible for sending the missing signal to the Signalling Termination.

6.2.1.2 Creation and deletions of BCSM instances

The Basic Call Controller Process creates an O_BCSM instance under the following conditions:

- Receipt of an Abstract Signalling Primitive from a Signalling Termination to initiate call establishment (i.e. Setup.Req),

- Receipt of an instruction from the SSF resulting from one of the following operations:

  - InitiateCallAttempt,

  - Connect in response to a TDP-R or EDP-R for a T_BCSM.

NOTE: At the SSF level, when only a controlling leg is left in a CS e.g. due to a SplitLeg operation, a BCSM instance is temporarily assigned to that controlling leg and is re-assigned to the passive leg(s) when connected again.

The Basic Call Controller creates a T_BCSM instance when it receives an indication from an O_BCSM instance that the PIC Send Call has been entered. After the T_BCSM has been created, the communication between the two instances uses the Inter-BCSM interface (IBI).

The Basic Call Controller Process deletes a BCSM instance under the following conditions:

-    Indication from the BCSM that the Null state has been reached;

-    Receipt of an instruction from the SSF resulting from a MergeCallSegments operation.

For each BCSM instance it creates, the Controller Process maintains the following knowledge:

-    the reference of the peer BCSM instance;

-    the reference of the associated Signalling Termination;

-    the reference of the underlying bearer resource (i.e. Bearer Termination).

Using these references, the BCC can determine how many, and which BCSM instances are associated with the same Signalling Termination.

6.2.2   Signalling Terminations

6.2.2.1   Overview

Signalling Terminations provide adaptation functionality between the Basic Call Controller and signalling interfaces. Different types of Signalling Terminations may be available in a CCF. This depends on the list of signalling protocols supported by the exchange in which the CCF resided.

Typical types of signalling terminations are:

- Analogue;

- ITU-T Recommendation Q.763 ISUP;

- ITU-T Recommendation Q.931 DSS.1;

- ITU-T Recommendation Q.2763 B-ISUP;

- ITU-T Recommendation Q.2931 DSS.2;

- H.225;

- SIP/SDP;

- BICC.

    NOTE: The signalling termination for analogue interfaces requires access to the Bearer Controller in order to detect physical events such as off-hook/on-hook.

Each Signalling Termination embodies the knowledge of the signalling procedures defined for the supported protocol.

Signalling Termination instances are created when an external event is received or is to be sent, to initiate call establishment. In the latter case, the Basic Call Controller creates them.

Signalling Termination instances communicates with the Basic Call Controller Process, using a set of Abstract Signalling Primitives defined below.

With regards to the Signalling Terminations, the Basic Call Controller performs the following actions:

- receive Abstract Signalling Primitives from the BCSM instances and transmits them to the appropriate Signalling Termination;

- receive Abstract Signalling Primitives from the Signalling Terminations and transmits them to the appropriate BCSM instance or to the SSF (e.g. charging events, USI related events).

In case there is more than one BCSM instance associated with the same Signalling Termination, the Basic Call Controller Process relays the information received from the Signalling Termination, to the firstly created BCSM instance only.

Signalling Termination instances use a Finite State Machine (FSM) to determine whether an Abstract Signalling Primitive, received from the BCSM Controller should be ignored or mapped to an appropriate NNI/UNI signalling message.

EXAMPLE: In case a BCSM sends a "Setup.Resp" Abstract Signalling Primitive to an ISUP Termination, the FSM state would determine whether this signal has to be mapped to an ANM message or ignored (because the FSM is already in the state "Confirmed Path" due to a follow-on call configuration).

6.2.2.2   Abstract Signalling Primitives

As shown in figure 3, the communication between the BCSM instances and the Signalling Terminations uses a set of Abstract Signalling Primitives defined in the following paragraphs. Similar Abstract Signalling Primitives are also used for inter-BCSM communication over the Inter-BCSM Interface (IBI).
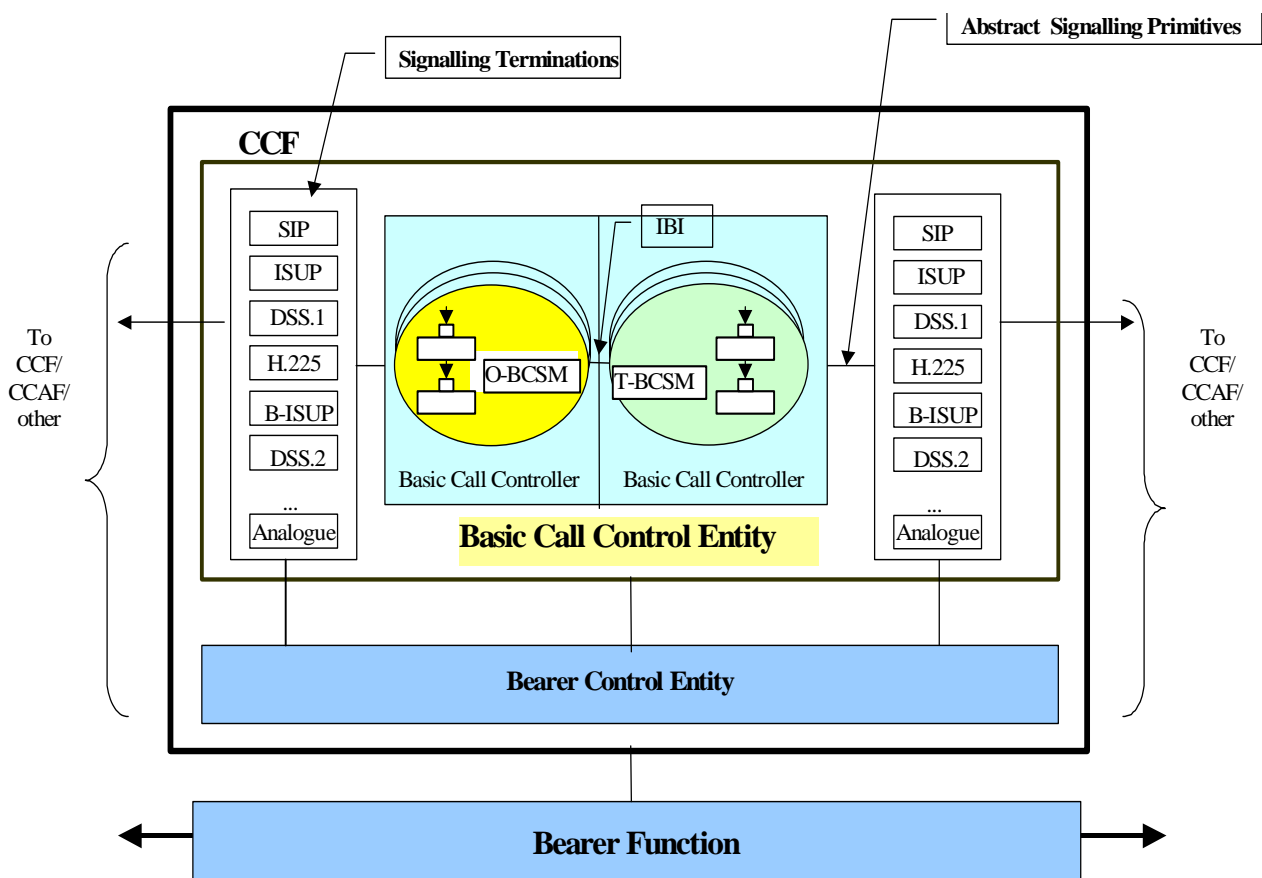
Figure 3: Basic Call Control Entity

The communication between the BCSM instances and the Signalling Termination uses a set of Abstract Signalling Primitives. Similar Abstract Signalling Primitives are also used for inter-BCSM communication (i.e. over the IBI interface). These Abstract Signalling Primitives are derived from the set of abstract services described below, using standards conventions. As shown in figure 4, these conventions are extended for use over the IBI.
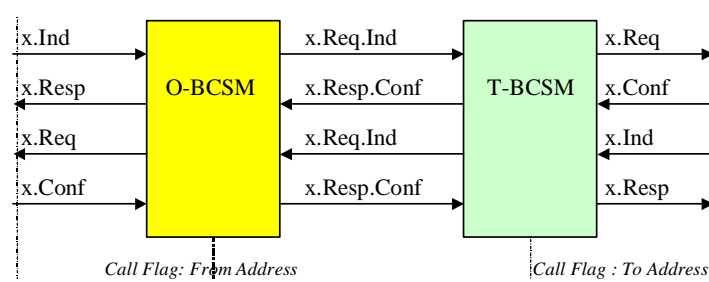
Figure 4: Abstract Signalling Primitive Conventions

Setup

The Setup service is used to request establishment of a call connection. This is a confirmed service, i.e. a response confirmation Setup Abstract Signalling Primitive is used to confirm that the connection has been established. Either the user or the network (i.e. SCF) can originate the request for establishment of a connection.

Release

The Release service is used to notify that a user has disconnected from the connection and cannot be connected and to request disconnection of a call connection. This is an unconfirmed service.

SubsequentAddress

The SubsequentAddress service is used for conveying subsequent address information during the digit-by-digit methods of call setup and for conveying information about last digit received, i.e. address end during the digit-by-digit methods of call setup. This is an unconfirmed service.

CallProgress

The CallProgress service is used to report status and/or other types of call information across the network. The type of information is indicated (e.g."no indication", "alerting", "remote call hold", etc.). This is an unconfirmed service.

NetworkSuspend

The NetworkSuspend service is a signal used to suspend the call on behalf of the called party upon receipt of an on-hook indication from the terminating line or upon receipt of a network suspend message indication from terminating side. This is an unconfirmed service.

NetworkResume

The NetworkResume service is used to resume the call on behalf of the called party upon receipt of a re-answer indication from the terminating line as the subscriber goes off-hook or upon receipt of a network resume message indication from terminating side. This is an unconfirmed service.

Failure

The Failure service is used to report the occurrence of a failure in the network.

ServiceFeature

The ServiceFeature service is used to report the occurrence of a service feature activation request from user.

ChargingEvent

The ChargingEvent service is used to report the occurrence of a charging event.

Data

The Data service is used to notify that a user has provided data and to request the transmission of data on a signalling connection. This is an unconfirmed bi-directional service.

6.2.2.3   Signalling Terminations

Within the CCF, Basic Signalling Terminations provide adaptation functionality between the Basic Call Controller and signalling interfaces. The concept of Signalling Terminations and its relationship to the other components of the CCF have been described previously.

Basic Signalling Termination instances use a **Finite State Machine (FSM)** to determine whether an Abstract Signalling Primitive, received from the BCSM Controller should be ignored or mapped to an appropriate signalling message. The actual mapping between Abstract Signalling Primitives and the messages from a particular signalling system is outside the scope of the present document.

6.2.2.4   Signalling Configurations

A set of Signalling Configurations is identified.
Annex D describes Signalling Configurations and illustrates the relationships between Signalling Configurations and CSCV states.

6.2.3   FSM for Call Control Signalling Termination

6.2.3.1   Overview

Figure 5 provides an overview of the states and transitions of the FSM for Call Control Signalling Terminations. These states and transitions are more precisely defined in the following clauses.
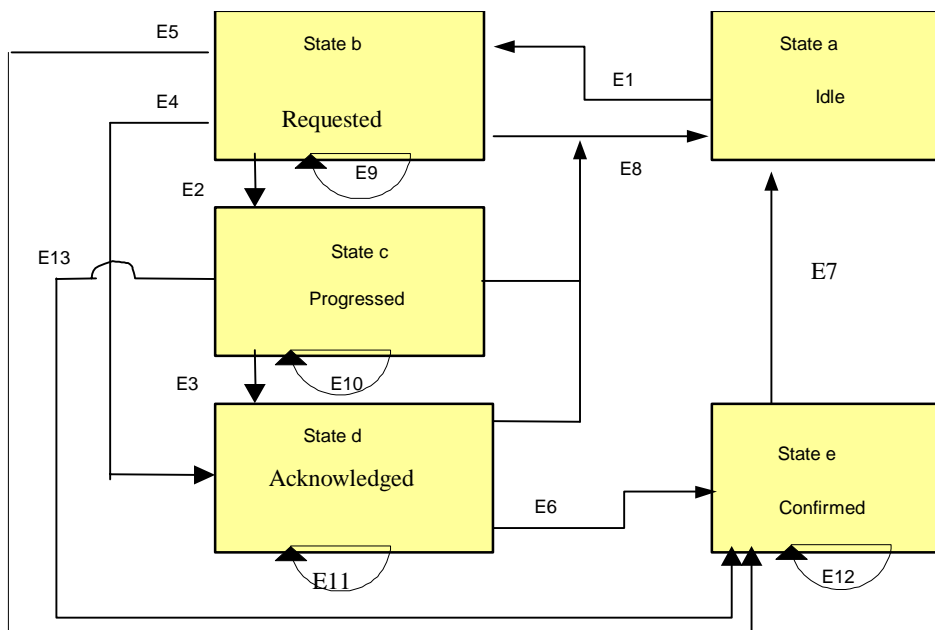
Figure 5: FSM for Call Control Signalling Termination

6.2.3.2   FSM states

6.2.3.2.1   Idle

The termination is not engaged in any call.

6.2.3.2.1.1   Entry events

See exit events from all other states.

The FSM instance is deleted when the release procedure has been completed.

6.2.3.2.1.2 Exit Events

- To the Requested Path state (E1):

  - Setup.Ind sent to an O_BCSM;

  - Setup.Req received from a T_BCSM.

6.2.3.2.2 Requested Path

Set-up request has been sent or received but not acknowledged nor confirmed.

6.2.3.2.2.1 Entry Event

- Setup.Ind sent to an O_BCSM;

- Setup.Req received from a T_BCSM.

6.2.3.2.2.2 Exit Event

- To the Idle state (E8):

  - Release.Ind sent to an O_BCSM or a T_BCSM;

  - Release.Req received from an O_BCSM or a T_BCSM;

  - Failure.Ind sent to an O_BCSM or a T_BCSM.

- To the Progressed Path state (E2):

  - CallProgress.Ind sent to a T_BCSM, with no alerting indication;

  - CallProgress.Req received from an O_BCSM with no alerting indication.

- To the Acknowledged Path state (E4):

  - CallProgress.Ind sent to a T_BCSM, with alerting indication;

  - CallProgress.Req received from an O_BCSM with alerting indication.

- To the Confirmed Path state (E5).

- Setup.Conf sent to a T_BCSM.

- Setup.Resp received from an O_BCSM.

The following Abstract Signalling Primitives may be received or sent without causing any state transition (E9):

- SubsequentAddress.

6.2.3.2.3 Progressed Path

Call progress information sent or received.

6.2.3.2.3.1 Entry Event

See exit events from Requested Path.

6.2.3.2.3.2 Exit Event

- To the Idle state (E8):

  - Release.Ind sent to an O_BCSM or a T_BCSM;

  - Release.Req received from an O_BCSM or a T_BCSM;

- Failure.Ind sent to an O_BCSM or a T_BCSM.

- To the Acknowledged Path state (E3):

    - CallProgress.Ind sent to a T_BCSM, with alerting indication;

    - CallProgress.Req received from an O_BCSM with alerting indication.

- To the Confirmed Path state (E13):

    - Setup.Conf sent to a T_BCSM;

    - Setup.Resp received from an O_BCSM.

The following Abstract Signalling Primitives may be received or sent without causing any state transition (E10):

- SubsequentAddress;

- ServiceFeatureIndication;

- ChargingEvent;

- Data.

6.2.3.2.4 Acknowledged Path

Alerting information sent or received.

6.2.3.2.4.1 Entry Event

See exist events from Progressed (E3) and Requested (E4).

6.2.3.2.4.2 Exit Event

- To the Idle state (E8):

    - Release.Ind sent to an O_BCSM or a T_BCSM;

    - Release.Req received from an O_BCSM or a T_BCSM;

    - Failure.Ind sent to an O_BCSM or a T_BCSM.

- To the Confirmed Path state (E6):

    - Setup.Conf sent to a T_BCSM;

    - Setup.Resp received from an O_BCSM.

The following Abstract Signalling Primitives may be received or sent without causing any state transition (E11):

- SubsequentAddress;

- ServiceFeature;

- ChargingEvent;

- Data.

6.2.3.2.5 Confirmed Path

Set-up conformation sent or received.

6.2.3.2.5.1 Entry Event

See exit events from the Requested state (E5) and Acknowledged state (E6).

6.2.3.2.5.2   Exit Event

- To the Idle state (E8):

    - Release.Ind sent to an O_BCSM or a T_BCSM;

    - Release.Req received from an O_BCSM or a T_BCSM;

    - Failure.Ind sent to an O_BCSM or a T_BCSM.

The following Abstract Signalling Primitives may be received or sent without causing any state transition (E12):

- SubsequentAddress;

- ServiceFeature;

- ChargingEvent;

- Data;

- NetworkSuspend;

- NetworkResume.

6.2.3.3   Example mapping

The following tables illustrate a possible mapping performed by an ISUP Signalling Termination.

EXAMPLE: Mapping Abstract Signalling Primitives to/from ISUP messages by originating Signalling Termination FSM.

Table 1

FSM Signalling Termination State ➔ Receipt of Abstract Signalling Primitive⬇	Idle	Requested	Progressed	Acknowledged	Confirmed
Setup.Ind	IAM Requested	/	/	/	/
Call Progress.Req (no alerting)	/	CPG Progressed	Same State	CPG Same state	Discard or CPG Same State
Call Progress.Req (alerting)	/	ACM Acknowledged	ACM Acknowledged	Discard	Discard or CPG Same State
Setup.Resp	/	ANM Confirmed	CON Confirmed	ANM Confirmed	Discard or CPG Same State
Release.Req	/	REL Idle	REL Idle	REL Idle	REL Idle

6.2.4   Signalling Interworking

6.2.4.1   General Objectives

1. **Signalling Transparency between call parties:**
   For the user initiated call involving 2 parties a signalling relation with signalling transparency (end-to-end signalling) shall be established between the calling and the called party in the original call setup in the same way as if no IN triggering had occurred on a basic 2-Party call. The signalling transparency is as specified in EN 301 070-1. This signalling transparency applies until the calling or called party disappears, i.e. as long as they exists in the call. This should also apply when the bearer connection is interrupted (e.g. due to SplitLeg). This is to secure support for end-to-end signalling and avoid degradation at interworking with other services in the call (e.g. ISDN supplementary service (e.g. UUS), PSS.1 (Q.sig), charging information).

No such end-to-end signalling transparency is defined for a) additional created legs using ICA and b) SCF initiated calls.

Depending on the underlying signalling network the signalling transparency may be broken or kept in case CPH operations are performed. If for example a splitleg operation is received the impact for a connection associated (facility associated) signalling system is that the information as well as the control plane will be impacted.

2. **Default Signalling Transparency:**
   The signalling transparency (end-to-end) is as default always kept between the incoming leg and the original created outgoing leg (basic two party call) independently whether those legs are exported (e.g. SplitLeg) or imported (e.g. MoveLeg) during a call.

   NOTE: Future enhancements (e.g. CS-4) could allow SCF to control for which legs the signalling transparency is to be established at call setup and so overrule this default. For example an indication in ICA could indicate that a basic 2-party call establishment (via ICA (Stable-1-Party) and MoveLeg to 'Originating_Setup') with end-to-end signalling transparency is requested.

3. **Follow-on call:**
   The signalling transparency (end-to-end) between the incoming leg and a subsequent created outgoing leg (none basic two party call) after the original call has been confirmed (answer has been sent) may have reduced signalling transparency as this call is established after the original call has been confirmed i.e.answer has been sent to the calling party for the first call. However, it shall at least be possible to propagate some information (e.g. APM including Q.SIG, charging information) between the original calling party and the new subsequent called (created) party in the follow-on call.

4. **Release handling:**
   Release handling is based upon the call release behaviour for a basic 2-party call, but extended to cater for multiple parties (legs). Defined rules specifies the release handling.
   Refer to "Functional Procedures for Call Segment (CS)" in clause 6.

6.3 Functional Interface between the CCF and the SSF

Within the CCF, the Basic Call Controller is the interface to the SSF. The following interactions may occur between these entities:

- CCF to SSF:

  - DP reporting;

  - BCSM deletion indication;

  - Processing errors in response to SSF instructions;

  - Report of signalling events that are not related to call control
    (i.e. receipt of a Data.ind Abstract Signalling Primitive);

  - Report of charging events received from a signalling termination (i.e. receipt of a ChargingEvent.ind Abstract Signalling Primitive);

  - Report of charging events resulting from the local charging GENeration function.

- SSF to CCF:

  - Creation of a BCSM instance;

  - Deletion of a BCSM instance;

  - Instructions to the charging GENeration function;

  - Instructions to resume call processing in sequence with the current DP (PICResume);

  - Instructions to resume call processing at a particular PIC (PICInit);

  - Instructions to the signalling terminations requesting the sending of messages that are not related to call control.

6.4   Basic call manager (BCM)

A brief description of the BCM component in the CCF/SSF model is provided in EN 301 931-1.
It provides an abstraction of a part of a switch that implements basic call and connection control to establish communication paths for users and to interconnect such communication paths.

The particular BCM subjects addressed below include the basic call state model (BCSM), basic call and connection events that can lead to the invocation of IN service logic instances, and basic call and connection events that should be reported to active IN service logic instances.

The BCM manages CCF/SSF resources required to support basic call and connection control.
The BCM also implements the O/T_BCSM and the DP processing. The DP processing is the entity of the BCM that interacts with the FIM as described in the FIM/CM description.

A high-level description of these subjects is provided below.

6.4.1   BCSM Model

In the present document, the BCSM provides a high-level model description of CCF activities required to establish and maintain communication paths for users. As such, it identifies a set of basic call and connection activities in a CCF and shows how these activities are joined together to process a basic call and connection (i.e. establish and maintain a communication path for a user).

Many aspects of the BCSM are not externally visible to IN service logic instances. However, aspects of the BCSM that are reflected upward to the IN-SM and FIM/CM are visible to IN service logic instances. Only these aspects of the BCSM will be the subject of standardization. As such, the BCSM is primarily an explanatory tool for providing a representation of CCF activities that can be analysed to determine which aspects of the BCSM will be visible to IN service logic instances, if any, and what level of abstraction and granularity is appropriate for this visibility.

The BCSM identifies points in basic call and connection processing when IN service logic instances are permitted to interact with basic call and connection control capabilities.
In particular, it provides a framework for describing basic call and connection events that can lead to the invocation of IN service logic instances or should be reported to active IN service logic instances, for those points in call and connection processing at which these events are detected, and for describing those points in call and connection processing when the transfer of control can occur.

Figure 6 shows the components that have been identified to describe a BCSM, to include: points in call (PICs), detection points (DPs), BCSM transitions, and events. PICs identify CCF activities associated with one or more basic call/connection states of interest to IN service logic instances. DPs indicate states in basic call and connection processing at which transfer of control from non-IN to IN service logic can occur. Other states may exist (e.g. within PICs) where call processing is suspended but IN service logic cannot be invoked.
BCSM transitions indicate the normal flow of basic call/connection processing from one PIC to another.
Entry events cause BCSM transitions into PICs. Exit events represent the result of PIC processing.
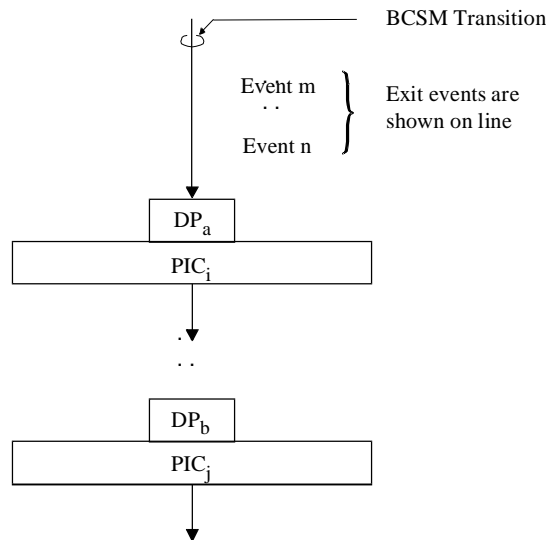
Figure 6: BCSM components

6.4.2   BCSM description

The BCSM described in this clause is based on the overall BCSM in annex A of ITU-T Recommendation Q.1204, ITU-T Recommendation Q.1238 and EN 301 140-5, refined as applicable.

The CCF/SSF model includes an originating half BCSM and a terminating half BCSM, each of which is managed by a functionally separate BCM in the CCF/SSF.
The description herein is to identify the aspects of the BCSM that are visible to IN service logic instances and the nature of the operations between the CCF/SSF and SCF.
It reflects the functional separation in the CCF/SSF model between the originating and terminating portions of calls that serves to isolate single-ended service logic instances related to the calling party from single-ended service logic instances related to the called party for the same call, i.e. the "half call" model approach.

The O_BCSM and T_BCSM respectively models existing switch processing of a basic two-party call with the extended transitions as needed in support of IN call capabilities including CPH.
In the following descriptions, the PICs are related at a high level to EN 300 403-1 ISDN call states. This is not intended to be a detailed formal definition of the relation between the PICs and EN 300 403-1 ISDN call states, but is intended as a point of reference to use in understanding the PICs. In particular, there are a number of possible ways in which the EN 300 403-1 call states may be traversed in certain situations that are considered below.

BCSM Information:
In order to enable independence between services offered during one call session when the PICs may be traversed several times, it is necessary at each PIC to maintain available a specific set of data until the calling (e.g. controlling) user releases and to ensure that software resources are returned to a coherent status when call processing passes through the PICs.

The BCSM Information depends on the applied signalling interface arrangement and is not listed in the following O/T_BCSM description. The general rule is that information that may be sent in the query (a service initiating operation as InitialDP) is to be maintained as long as an IN service could be triggered on the O/T_BCSM. More precise rules and mapping tables are provided in specific interworking documents; e.g. EN 301 070-1 and ITU T Recommendation Q.1601.

The information that is sent to the SCF at a given trigger detection point is a subset of the information described thereafter. Other information may be available at a given PIC that is not used by processing at the PIC or is only used by underlying call processing.

DP Naming:
In order to maintain uniqueness of DP names between the originating and terminating half BCSMs, "O" and "T" is prefixed to certain originating and terminating DP names, respectively.

For ease of reference, the DPs associated with the BCSM transition implied by each entry and exit event for each PIC are listed along with the PIC descriptions.

BCSM Transitions:

The BCSM description provided hereafter only describes the *basic transitions* which occur when processing a basic two party call.

A basic call signifies a call between two users that consists of communication only, and does not include additional features (ITU-T Recommendation Q.1290).

The PIC entry event is normally the Resume instruction received from the SSF after the processing of a DP: « basic » transition DP to PIC.

PIC exit events are the result of the processing of an event received from the signalling interface arrangement supported by IN: « basic » transitions PIC to DP/PIC.

The transitions which may occur due to an SCF instruction (« extended » transitions from DP to DP/PIC) and which may change the basic evolution of the call are not described here.

Later clauses describe the complete set (i.e. also the « extended » transitions from DP to DP/PIC) of possible BCSM transitions for the O_BCSM and the T_BCSM respectively.

In addition a mapping is required between signalling events and BCSM events, for each signalling interface arrangement supported by IN. In this respect it includes INAP as well as the NNI/UNI signalling arrangements.

Since the BCSM is generic, it may describe events that do not apply to certain signalling arrangements. It is important to understand and describe how each signalling arrangement applies to the BCSM.

NOTE: CCAF functionality is not explicitly modelled.

6.4.2.1   Originating BCSM

The originating half of the BCSM corresponds to that portion of the BCSM associated with the originating party.

Figure 7: Originating BCSM for IN CS-3: Basic Set of Transitions

The following information is available at all PICs in the O_BCSM:

- Calling Party Class of Service (where locally available e.g. subscriber profile including calling party category information).

- Call Reference (where locally assigned in the switch).

- Terminal Type - see ITU-T Recommendation Q.1290. The SCF uses this to determine the most appropriate form of user-interaction to use (e.g. in-band announcements). This information may only be available at originating local exchanges.

- Calling Party Facility Restriction Level (FRL).
  The permission level associated with an incoming facility, e.g. trunk line.

- Calling Party BGID (Business Group Identity).

- The Basic Business Group or Multiswitch Business Group Identity of the calling party, e.g. group-based services.

The above information applies in addition to Signalling Information received from the signalling interface and from the peer T_BCSM. The Signalling Information depends on the signalling interface arrangement applied: e.g. DSS.1, DSS.2, analogue user, conventional trunk, SS7 trunk and N-ISUP or B-ISUP, private trunk, SIP, H.225 etc.

The description for each of the PICs in the originating half of the BCSM is provided below.

NOTE: See also subsequent description on "BCSM Indications for the CS_3 Call Model" for more information concerning PICs.

6.4.2.1.1   O_Null

Entry event: Disconnect and clearing of a previous call (DPs: O_Disconnect and O_Abandon), or default handling of exceptions by CCF/SSF completed.

Functions: Interface (line/trunk e.g. DSS1/N-ISUP or DSS2/B-ISUP interface) is idled (no call exists, no call reference exists, etc.). Supervision is being provided.

Information available: After detecting the Origination Attempt event, it is assumed that the CCF/SSF has the information corresponding to the Signalling Information received from the signalling interface arrangement as a Setup.Ind Abstract Signalling Primitive is received (due to e.g. a DSS.1 Setup, DSS.2 Setup, N-ISUP IAM, B-ISUP IAM, SIP INVITE) available and associated with the originating call portion. If the CCF/SSF determines that the call origination attempt is denied, the cause of the failed authorization is also known. The above information applies in addition to the information available at all PICS in the O_BCSM.

Exit event:

- Indication of desire to place an outgoing call by the receipt of a Setup service request primitive. This event causes call processing to move to the Origination_Attempt DP.

Corresponding EN 300 403-1 call state: O_Null

NOTE: The O_Abandon event occurs when the originating party abandons the call before the call has been answered, i.e. when the calling party disconnects and a Release.Ind Abstract Signalling Primitive is received.

For example, this event can result from one of the following:

- the CCF/SSF receives an on-hook indication from a caller served by a non-ISDN line, following switch-hook flash timing;

- the CCF/SSF receives a call clearing message from a caller served (e.g. by a narrowband or broadband ISDN interface (DDS1/DSS2));

- the CCF/SSF receives a disconnect indication (e.g. from a conventional trunk or private facility trunk);

- the CCF/SSF receives a Release Message (e.g. from an SS7 trunk, N-ISUP or B-ISUP interface).

6.4.2.1.2   Authorize_Origination_Attempt

Entry event: An indication is available that the originating terminal needs to be authorized. (DP: Origination_Attempt).

Functions:

- The originating terminal rights should be checked using the calling party's identity and service profile. The authority/ability of the party to place the call with given properties (e.g. bearer capability, line restrictions) is verified.
  The types of authorization to be performed may vary for different types of originating resources (e.g. for lines vs. trunks).

- Other features which might be required during this PIC are not described.

Information available: After detecting the Origination_Attempt_Authorized event, it is assumed that the CCF/SSF has the same information available associated with the originating portion of the call as it did after detecting the Origination_Attempt event in the O_Null PIC.

- Information as per the O_Null PIC.

- Authorization result - If the CCF/SSF determines that the origination attempt is denied, the cause of the failed authorization is also known.

Exit event:

- An indication is received that the authorization is successful. This event causes call processing to move to the Origination_Attempt_Authorized DP.

- The O_Abandon event occurs when an indication of originating party abandon is received, i.e. a Release.Ind Abstract Signalling Primitive is received. This event causes call processing to move to the O_Abandon DP.

- An indication is received that the call origination is denied. The call processing moves to the Origination_Attempt_Denied DP.

6.4.2.1.3   Collect_Information

Entry event: Authority/ability to place outgoing call verified. (DP: Origination_Attempt_Authorized).

Functions:

- Initial information package/dialling string (e.g. service codes, prefixes, dialled address digits) being collected from originating party. Information being examined according to dialling plan to determine end of collection. No further action may be required if an en bloc signalling method is in use (e.g. en bloc signalling used by an ISDN terminal or an incoming SS7 trunk).

- The CCF/SSF shall be able to support subsequent digit collection according to trigger criteria assigned before sending the query. For example, if a feature code (e.g. *64) is entered, the CCF/SSF may:

    - collect digits according to the normal dialling plan; or

    - collect a variable number of digits.

Information available: After the CCF/SSF determines that information collection is complete, it is assumed that the CCF/SSF has the following information available and associated with the originating portion of the call:

- Information as per the O_Null PIC.

- Collected Information - As obtained from the Setup service request and possible SubsequentAddress:Ind Abstract Signalling Primitives. This is illustrated in the **examples** below.

The Collected Information may for example consist of one or more of the following:

- Access Codes within a Customized Dialling Plan (CDP) - see ITU-T Recommendation Q.1290.

    The Customized Dialling Plan (CDP) in force may specify that after a given access code is dialled, more digits are to be collected according to the "normal dialling plan", i.e. the dialling plan in force. In this case, Access Code and Collected Address Information are known. If the CDP in force specifies that after a given access code is dialled, a variable number of digits are to be collected, then Access Code and Collected Digits are known.

- Feature Code - see ITU-T Recommendation Q.1290 and ITU-T Recommendation Q.762 Feature Code Signalling Information where this parameter is defined for national use only.

- If the numbering plan in force specifies that after a given feature code is dialled, more digits are to be collected according to the "normal dialling plan", then Feature Code and Collected Address Information are known. If the dialling plan in force specifies that after a given feature code is dialled, a variable number of digits are to be collected, then Feature Code and Collected Digits are known. The service associated with the feature code is dependent upon the users service profile.

- Facility Code - see ITU-T Recommendation Q.1290. This information may be provided if and when facility selective service signalling is supported.

- Feature Activation - see ITU-T Recommendation Q.932 Feature Activation information element.
  If the CDP in force specifies that after a given feature activator is received, more digits are to be collected according to the numbering plan, then Feature Activation Indicator and Collected Address Information are known.
  If the CDP in force specifies that after a given feature activator is received, a variable number of digits are to be collected, then Feature Activation Indicator and Collected Digits are known.

- Prefix.

- Carrier Access Code/Carrier Identification Code - see ITU-T Recommendation Q.1290.
  The caller may dial a Carrier Access Code (CAC) (e.g. a 101XXXX for use on this call). When the caller is served by an ISDN interface, a Carrier Identification Code, i.e. XXXX, may be received by the CCF/SSF within the transit network selection information element of the ISDN SETUP message.

- Collected Address Information - see ITU-T Recommendation Q.1290.
  Available as per the numbering plan.

- Numbering Plan Indicator - see ITU-T Recommendation Q.762 Numbering Plan Indicator signalling information.

- Collected Digits - see ITU-T Recommendation Q.1290.
  The numbering plan in force may specify that after a given Feature Activation, Feature Code, or Access Code within a CDP is dialled, a variable number of digits are to be collected using normal inter-digit timing. In this case, these collected digits are also known at this time.

- Numbering Plan Indicator - see ITU-T Recommendation Q.762 Numbering Plan Indicator signalling information. The address received is expected to conform to E.164.

- Carrier Selection - see ITU-T Recommendation Q.1290. –   Originating Line Information -
  see ITU-T Recommendation Q.1290.
  This information is only known when MF signalling is used on the originating trunk (network operator specific). In this case, the Originating Line Information is sent during the second stage of overlap out pulsing.

Exit events:

- Availability of complete initial information package/dialling string from originating party. (This event may have already occurred in the case of en bloc signalling, in which case the waiting duration in this PIC is zero).
  This event causes call processing to move to the Collected_Information DP.

- The following exception exit events are applicable to this PIC: Collect_Failure. The Collect_Failure event encompasses events such as CollectTimeout, CollectInfoFailure and InvalidInformation.

  - The CollectTimeout event is detected when enough information to process the call was not received by the CCF/SSF before a normal interdigit timer expires. For example for an SS7 trunk, this event corresponds to the IAM not containing the information necessary to process the call. In this case there may be no timing involved (timing may be involved for ISUP overlap sending).

  - The CollectInfoFailure event is detected when the CCF/SSF is unable to perform the information collection due to a lack of switch resources (e.g. no digit receivers are available).

  - The Invalidinformation event occurs when the information received from the caller is not valid, for instance the information received violates the dialling plan in force.

  - The O_Abandon event occurs when an indication of originating party abandon is received, i.e. a Release.Ind Abstract Signalling Primitive is received. This event causes call processing to move to the O_Abandon DP.

Comment: Some digit analysis is required to determine the end of dialling. However, it is assumed that this analysis may be modelled as separable from the rest of digit analysis, which occurs in the Analyse_Information PIC. There is no intention to specify an implementation. However, a switch should externally present the separable view described for closed numbering plans (see note 2).

In the case a complete number (e.g. ISDN en bloc sending) is received in a Setup.Ind Abstract Signalling Primitive, call processing passes to the Collected_Information DP.

NOTE 1: The BCSM transits to Collected_Information DP when the initial information package/dialling string is received from the calling party - this occurs when enough information is received to proceed with call processing (e.g. as in the case of ISDN overlap sending or MF out pulsing). Specifically, for the digit by digit collection case (i.e. receipt of SubsequentAddress.Ind Abstract Signalling Primitive), if the Collected_Information DP is armed as a Trigger Detection Point-Request (TDP-R), the SSF sends the query, i.e. an InitialDP operation to the SCF when enough digits is received to determine if the TDP criteria is met. It suspends BCSM processing but will collect further digits. It is network operator specific to determine when complete information is available (see note 2).

NOTE 2: This separable view is provided by supporting distinct DPs. The Collected_Information DP is used after digit collection and the Analysed_Information DP is used after the rest of the digit analysis.

NOTE 3: In some networks, it may be not possible for the CCF/SSF to determine when the called number information is complete. Therefore, TDP criteria for Collected_Information DP may be met in such networks before the called number information is complete.

Corresponding EN 300 403-1 call state: 1. Call Initiated and (optionally) 2. Overlap Sending.

6.4.2.1.4   Analyse_Information

Entry event: Availability of complete initial information package/dialling string from originating party
(DP: Collected_Information) or Route_Busy event reported from the Select Route PIC.

Function: Information being analysed and/or translated according to dialling plan to determine routing address and call type (e.g. local exchange call, transit exchange call, international exchange call).

One of the results of processing in this PIC is determination of routing address:

i)   called party number only (called party number is served by the SSP);

ii) called party number and route index, where the route index is a pointer to a trunk group to route an out going call attempt on (called party number is served by another exchange);

iii) called party number and route index, where the route index is a pointer to a list of trunk groups to route an outgoing call attempt on (called party number is served by another exchange).

Information available: After the CCF/SSF determines the information has been analysed, it is assumed that the CCF/SSF has the following information is available and associated with the originating portion of the call:

- Information as per the O_-Null PIC.

- Analysis Results (of the Collected Information) - as described in the **examples** below.

The Analysis Results consists of for example one or more of the following:

- Called Party Number.

- Numbering Plan Indicator.
  See ITU-T Recommendation Q.762 Numbering Plan Indicator signalling information.

- Type Of Call - see ITU-T Recommendation Q.1290.

- Carrier - see ITU-T Recommendation Q.1290.

- Carrier Identification Code - see ITU-T Recommendation Q.1290.
  Available for Internetwork carrier calls.

- Carrier Selection - see ITU-T Recommendation Q.1290.
  Available for Inter Serving Area ID carrier calls.

- Collected Information - Access Code within a CDP, Feature Code, Feature Activation, Prefix, Carrier Access Code/Carrier Identification Code, Collected Address Information/Digits - as described under the Collect_Information PIC.

- Originating Line Information - see ITU-T Recommendation Q.1290.
  Available for Inter Serving Area ID carrier calls.

- Route Index - see ITU-T Recommendation Q.1290.
  Available if this call does not terminate on this CCF/SSF.

Exit events:

- Availability of routeing address and nature of address. This event causes call processing to move to the Analysed_Information DP.

- The following exception exit event is applicable to this PIC: InvalidInformation.
  The InvalidInformation event occurs when the information received from the caller is not valid and no further treatment (like e.g. routing of the call to an announcement in case of a wrong dialled number) can be applied to the call, for instance in case the information received violates the dialling plan in force.

- The O_Abandon event occurs when an indication of originating party abandon is received, i.e. a Release.Ind Abstract Signalling Primitive is received. This event causes call processing to move to the O_Abandon DP.

Comments: Note that routing address does not necessarily mean that the final physical route has been determined (e.g. route list has not been searched, hunt groups have not yet been searched, directory number has not yet been translated to physical port address), though this may be the case (e.g. when routing to a specific private facility).

Corresponding EN 300 403-1 call state: Not applicable.

6.4.2.1.5   Select Route

Entry events: Availability of routing address and call type. (DP: Analysed_Information) or route busy event reported from the Send_Call or O_Alerting PICs.

Functions:

- Routeing address and call type are interpreted.
  The next route is selected. This may involve sequentially searching a route list, translating a directory number into physical port address, etc. The individual destination resource out of a resource group (e.g. a multi-line hunt group, a trunk group) is not selected. In some cases (e.g. an analogue line interface), a single resource (not a group) is selected.

- When the entry event is the Route_Failure event from the Send_Call PIC (see below), the CCF/SSF must first check the Route Failure Condition 1, Route Failure Condition 2, or Route Failure Condition 3 as defined under the Send_Call PIC exit events. If these conditions are true, then the call shall proceed to the Analyse Information PIC by means of the Route_Busy event.

Depending on the location in the network where the route is busy, one of the following actions apply:

- Route_Busy event: If the trunk group selected for the call is busy at this switch, the CCF/SSF attempts to route the call on the next trunk group that has been specified for the call (when a route list is being searched or alternate routes are specified by the SCF). Call processing moves to the Analyse_Information PIC when one of two conditions occurs: 1) all private-facility trunk groups have been tried and routing over a public facility is allowed, or 2) routing to a particular intra or internetwork carrier has been tried and an alternate carrier is allowed.

- Route_Failure event: Call processing moves to the Route_Select_Failure DP when one of the two conditions occurs 1) all of the trunk groups (private and public) have been tried and no route is available or 2) route busy is detected at another switch (an indication of this condition may be received via a Release.Req.Ind Abstract Signalling Primitive).

Information available: After the CCF/SSF determines the route has been selected, it is assumed the CCF/SSF has the following information available and associated with the originating portion of the call:

- Information as per the O_Null PIC.

- Analysis Results - See description in the Analyse_Information PIC.

- Routing Information - When more than one route has been specified for the call (either by the SCF or as part of the information stored at the CCF/SSF), the CCF/SSF remembers what routes have been tried for this call and which route to select next. If the call is to an Inter Serving Area ID carrier, the routing information includes Circuit Code information.

Exit events:

- Route_Selected event: Route selection is successful and call processing moves to Authorize_Call_Setup PIC.

- Route_Select_Failure event: Unable to select a route (e.g. unable to determine a correct route, no more routes on route list) or indication from the terminating half BCSM via. a Release.Req.Ind Abstract Signalling Primitive that the call cannot be presented to the terminating party (e.g. network congestion).This event causes call processing to move to the Route_Select_Failure DP. The event indication received from T_BCSM that causes call processing to move to the O_Route_Select_Failure DP depends upon the event reason (cause value) as defined according to table 6.

- The Route_Busy event leading to the Analyse_Information PIC, or the following conditions are met:

  a) unable to select a route (e.g. unable to determine a correct route, no more routes on route list) or indication from the terminating half BCSM that call cannot be presented to the terminating party (e.g. network congestion);

  b) the route was determined by switch translations at the Analyse_Information PIC.

- The O_Abandon event occurs when an indication of originating party abandon is received, i.e. a Release.Ind Abstract Signalling Primitive is received. This event causes call processing to move to the O_Abandon DP.

6.4.2.1.6   Authorize_Call_Setup

Entry events: Route Selected event.

Function: The authority of the calling party to place this particular call is verified.

Information available: After the CCF/SSF determines the call setup has been authorized, it is assumed the CCF/SSF has the following information available and associated with the originating portion of the call with restrictions as noted:

- Information as per the O_Null PIC.

- Analysis Results - see description in the Analyse_Information PIC.

- Routing Information - see description in the Select_Route PIC.

Exit events:

- Call Setup Authorized event. The Call Setup Authorized event occurs when the authority to place the call is verified. For example for an SS7-supported trunk interface, if the received IAM indicates that a continuity check is being performed on the call connection and the call terminates to a non-ISDN line or ISDN interface, the Call Setup Authorized event occurs when an ISUP Continuity Message (COT) with a successful indication is received.

- The O_Abandon event occurs when an indication of originating party abandon is received, i.e. a Release.Ind Abstract Signalling Primitive is received. This event causes call processing to move to the O_Abandon DP.

- The Authorization Failure event occurs when the authority to place the call is denied (e.g. business group restriction mismatch, toll restricted calling line). For example for an SS7-supported trunk interface, the Authorization Failure event occurs when the continuity check procedure results in failure. This event causes a BCSM transition to the Authorize_Route_Failure DP.

6.4.2.1.7   Send_Call

Entry events: Call Setup Authorized event.

Functions: The CCF/SSF sends an indication of the desire to set up a call to the specified Called Party ID to the terminating call portion via a Setup.Req.Ind Abstract Signalling Primitive.

The information that may be passed to the terminating call portion is for example:
Calling Party ID; Calling Party BGID; Calling Party Category (determined by the Class of Service information or ISUP originating line information parameter); Bearer Capability; Called Party ID; Calling Party Subaddress; Called Party Subaddress; ForwardGVNS, Carrier; Route Index; Carrier Identification Code. Circuit Code, and Carrier Selection;.
Other feature-information not used by the processing modelled by this PIC (e.g. call forwarding, generic name, and business group information) may also be passed to the terminating call portion.

Information available: After the CCF/SSF determines the call has been delivered (to the terminating half), it is assumed the CCF/SSF has the following information available and associated with the originating portion of the call:

- Information as per the O_Null PIC.

- Analysis Results - see description in the Analyse_Information PIC.

- Routing Information - see description in the Select_Route PIC.

- Feature Activation Information - see description below:
  A service feature request, i.e. a ServiceFeature.Ind Abstract Signalling Primitive is received from the originating party (e.g. hook-flash, ISDN feature activator, DTMF provided control code).

Exit events:

- A Route_Failure event is detected when:

  i)   the following conditions are met, here after called **Route_Failure Condition 1:**

     a) an indication of a T_Busy event specifying route busy (received when the route at the local switch is found to be busy) is received from the terminating call portion via a Release.Req.Ind Abstract Signalling Primitive (presentation failure event from the Present Call PIC);

     b) the route was determined by switch translations at the Analyse_Information PIC;

  ii) the following conditions are met, here after called **Route_Failure Condition 2:**

     a) a "Call Rejected" event specifying route busy (received when the route is found to be busy at a switch other than the local switch) is received from the terminating call portion via a Release.Req.Ind Abstract Signalling Primitive (presentation failure event from the Present Call PIC);

     b) the route was determined by the switch translations at the Analyse Information PIC;

  iii) the following condition is met, here after called **Route_Failure Condition 3:**

     O_No_Answer event occurs.

The detection of the Route_Failure event causes the originating call portion to return to the Select_Route PIC.

After all Route_Failure events have been detected call processing moves to the O_Route_Select_Failure DP or O_Called_Party_Busy DP (e.g. User Busy, Subscriber absent (not reachable) or O_No_Answer DP or O_Exception depending upon the event reason (cause value) as defined according to table 6.

- An O_Answer event occurs when an indication of a T_Answer event is received from the terminating call portion via a Setup.Resp.Conf Abstract Signalling Primitive. This event causes call processing to move to the O_Answer DP.

- An O_Term_Seized event occurs when an indication of a call accepted event is received, i.e. a CallProgress(Alerting).Req.Ind Abstract Signalling Primitive is received from the terminating call portion. This event causes call processing to move to the O_Term_Seized DP.

- A service feature request event, i.e. a ServiceFeature.Ind Abstract Signalling. is detected from the originating party: e.g. hook-flash, ISDN feature activator, DTMF provided control code. This event causes call processing to move to the O_Mid_Call DP.

No additional actions are taken in a non-ISDN line or private facility trunk.

In these cases, audible ringing, if applicable, is being sent from the originating call portion of the terminating switch.

- The O_No_Answer event is an IN event or signalling event via a Release.Req.Ind Abstract Signalling Primitive. It can only occur when an O_No_Answer trigger is assigned and detected or when requested by a RequestReportBCSMEvent. If the O-No_Answer timer expires or an indication of the T_No_Answer event is received before an O_Answer event is detected (i.e. before the called party answers), the CCF/SSF reports the event to the SCF. The event indication received that causes call processing to move to the O_No_Answer DP depends upon the event reason (cause value) as defined according to table 6.

- The O_Called_Party_Busy event occurs when an indication of a T_Busy event specifying user busy is received via a Release.Req.Ind Abstract Signalling Primitive from the terminating portion of the call (e.g. network-determined-user-busy, user not reachable). This event also occurs when an indication of a "Call Rejected" event specifying user busy (i.e. user-determined-user busy) is received from the terminating portion of the call. For calls originating from non-ISDN lines, conventional trunks, and private-facility trunks, if an indication of busy is received from the terminating portion of the call and no originating triggers or requested events apply, busy tone is provided.

    In addition to these busy events, "Call Rejected" conditions are also treated as O_Called_Party_Busy events.

    In this case, the terminating portion of the call is cleared.
    The events that causes call processing to move to the O_Called Party_Busy DP depending upon the event reason (cause value) as defined according to table 6.

- For SS7-supported trunk interface, the Authorization_Route_Failure event occurs when the continuity check procedure results in failure. This event causes a BCSM transition to the O_Exception.

- The O_Abandon event occurs when an indication of originating party abandon is received, i.e. a Release.Ind Abstract Signalling Primitive is received. This event causes call processing to move to the O_Abandon DP.

6.4.2.1.8   O_Alerting

Entry event: O_Term_Seized event (DP: O_Term_Seized).

Function:

- Wait for the terminating party to answer. At this point, the caller receives in-band audible ringing (from the terminating switch).

- An indication of a call progress event CallProgress (Alerting).Req.Ind Abstract Signalling Primitive may be received from the terminating call portion. This may result in a call progress indication being sent backward.

Information available: When the CCF/SSF is in this PIC, it is assumed the CCF/SSF has the following information available and associated with the originating portion of the call:

- Information as per the O_Null PIC.

- Analysis Results - see description in the Analyse_Information PIC.

- Routing Information - see description in the Select_Route PIC.

- Feature Activation Information - A ServiceFeature.Ind Abstract Signalling Primitive is received from the originating party or a ServiceFeature.Req.Ind Abstract Signalling Primitive received from the terminating party e.g. hook-flash, ISDN feature activator, DTMF provided control code.

Exit events:

- The O_Answer event occurs when an indication of a T_Answer event is received from the terminating portion of the call via a Setup.Resp.Conf Abstract Signalling Primitive (e.g. terminating party goes off hook, *EN 300 403-1* Connect message received, ISUP Answer message received).
  This event causes call processing to move to the O_Answer DP.

- A ServiceFeature.Ind Abstract Signalling Primitive is detected from the originating party: e.g. hook-flash, ISDN feature activator, DTMF provided control code. The detection of this event causes call processing to move to the O_Mid_Call DP.

- A Route_Failure event is detected via a Release.Req.Ind Abstract Signalling Primitive and the following conditions are met:

  - O_Called_Party_Busy event, Route_Failure event or O_No_Answer event occurs.

  - The detection of the Route_Failure event causes the originating call portion to return to the Select_Route PIC.

- After all Route_Failure events have been detected call processing moves to the O_Route_Select_Failure DP or O_Called_Party_Busy DP (e.g. User Busy, Subscriber absent (not reachable) or O_No_Answer DP or O_Exception depending upon the event reason (cause value) as defined according to table 6.

- The O_No_Answer event from this PIC is the same as the O_No_Answer event defined as an Exit Event from the Send_Call PIC. I.e. the event indication received that causes call processing to move to the O_No_Answer DP depends upon the event reason (cause value) as defined according to table 6.

- From this PIC, the O_Called_Party_Busy event occurs either when:

  i) a Call Rejected event specifying user busy is received; or

  ii) when an indication of a Call Rejected event not specifying busy is received from the terminating call portion (as described in the Send_Call PIC).

  In addition, for a call to an ISDN user, after the SETUP message is offered and an ALERTing message has been received (i.e. the terminating call portion is in the T_Alerting PIC), the ISDN user may reject the call. This Call Rejected event is treated as an O_Called_Party_Busy event by the originating call portion.
  (DP: O_Called_Party_Busy).
  The events that causes call processing to move to the O_Called Party_Busy DP depends upon the event reason (cause value) as defined according to table 6.

- The O_Abandon event occurs when an indication of originating party abandon is received, i.e. a Release.Ind Abstract Signalling Primitive is received. This event causes call processing to move to the O_Abandon.

6.4.2.1.9   O_Active

Entry event: Indication from the terminating half BCSM that the call is accepted and answered by terminating party. (DP: O_Answer).

Function: In this PIC several processes may be initiated e.g.:

- Connection established between originating and terminating party. Message accounting/charging data may be being collected. Call supervision is being provided.

- The called party may be put on hold and returned to the active phase by a service logic.

- In case that a disconnect indication is received via a Release.Ind Abstract Signalling Primitive from the originating side or a Release.Req.Ind Abstract Signalling Primitive from the terminating side (T_BCSM), this PIC is immediately moved to the O_Disconnect DP without any action.
  As an option, the call can be continued for an appropriate period (retention timer) in order to activate a service or feature request from an originating or terminating party-on the call initiated via a ServiceFeature.Ind/ServiceFeature.Req.Ind Abstract Signalling Primitive.

Information available: Once the CCF/SSF has received an indication from the terminating half BCSM that the call has been answered, it is assumed the CCF/SSF has the following information available and associated with the originating portion of the call:

- Information as per the O_Alerting PIC.

- Feature Activation Information - A ServiceFeature.Ind Abstract Signalling Primitive is received from the originating party or a ServiceFeature.Req.Ind Abstract Signalling Primitive received from the terminating party e.g. hook-flash, ISDN feature activator, DTMF provided control code.

Exit events:

- A service or a feature request is received from an originating or terminating party, i.e. a ServiceFeature.Ind or a ServiceFeature.Req.Ind Abstract Signalling Primitive (e.g. DTMF provided control code, hook flash, ISDN feature activator, EN 300 403-1 HOLD or RETRIEVE message from originating party only). The detection of this event causes call processing to move to the O_Mid_Call DP.

- A disconnect indication is received from the terminating party via a NetworkSuspend.Req.IndAbstract Signalling Primitive (non-ISDN subscriber) via the terminating half BCSM. This event causes call processing to move to the O_Suspend DP. A disconnect timing is associated with this BCSM transition.

- A disconnect indication is received from the originating party or terminating party via a Release.Ind or a Release.Req.Ind Abstract Signalling Primitive. This event causes call processing to move to the O_Disconnect DP.

- A connection failure occurs (O_Exception).

Comments:

- If originating party abandons call, i.e. a Release.Ind Abstract Signalling Primitive is detected, while suspended at O_Answer DP a transition to DP: O_Abandon shall occur.

- Disconnect treatment and timing is different for call attempts originating from ISDN and analogue line interfaces, e.g. at a release in the ISDN network from an ISDN line a transition direct to O-Disconnect DP occurs.

Corresponding EN 300 403-1 call state: 10. Active

EN 300 403-1 call states corresponding to disconnect: 11. Disconnect request, 12. Disconnect indication and 19. Release request.

6.4.2.1.10   O_Suspended

Entry event: A NetworkSuspend.Req.Ind Abstract Signalling Primitive is received from the T_BCSM when the terminating party has disconnected (e.g. on-hook) (DP: O_Suspend).

Function:

- The connection between the originating and terminating party is maintained and depending on the incoming network connection, appropriate backward signalling takes place.

  - In case that a disconnect indication is received via a Release.Req.Ind Abstract Signalling Primitive from the T_BCSM, this PIC is immediately moved to the O_Disconnect DP without any action.
    As an option, the call can be continued for an appropriate period (retention timer) in order to activate a service or feature request from an originating or terminating party-on call initiated via a ServiceFeature.Ind/ServiceFeature.Req.Ind Abstract Signalling Primitive.

  - If the re-answer indication, from the T_BCSM is received via a NetworkResume.Req.Ind Abstract Signalling Primitive, the originating and terminating parties are reconnected.

Information available: It is assumed the CCF/SSF has the following information available and associated with the originating portion of the call:

- Information as per the O_Active PIC.

Exit event:

- Connection to the terminating party is resumed via a NetworkResume.Ind Abstract Signalling Primitive. The O_BCSM returns to the O_Active PIC. This event causes call processing to move to the O_Re-answer DP.

- A service feature request is received, i.e. a ServiceFeature.Ind Abstract Signalling from the originating party, e.g. hook flash, DTMF provided control code, ISDN feature activator of facility. The detection of this event causes call processing to move to the O_Mid_Call DP.

- A disconnection indication is received via a Release.Ind Abstract Signalling Primitive from the originating party. This event causes call processing to move to the O_Disconnect DP.

- A disconnection indication is received via a Release.Req.Ind Abstract Signalling Primitive from the terminating party.
  This event causes call processing to move to the O_Disconnect DP.

- An indication of expiration of the timer waiting for re-answer request is received from the T_BCSM. This event causes call processing to move to the O_Disconnect DP.

- A trigger at O_Mid_Call is not initiated during an appropriate period (DP: O_Disconnect).

- An exception event is encountered (O_Exception).

NOTE 1: A Call Retention timer may exist. Disconnect treatment and timing is different for call reconnection, call suspension and call retention.

NOTE 2: After the release of the outgoing connection, the originating party may initiate another call, e.g. a follow-on call.

6.4.2.1.11   O_Exception

Entry event: An exception condition is encountered (as described above for each PIC).

Function: Default handling of the exception condition is being provided. This includes general actions necessary to ensure no resources remain inappropriately allocated, such as:

- If any relationships exist between the SSF and SCF(s), send an error information to the SCF(s) closing the relationships and indicating that any outstanding call handling instructions will not run to completion (see note).

- If an SCF previously requested that call parameters be provided at the end of the call (see the CallInformationRequest operation), these should be included in the error information.

- The CCF/SSF should make use of vendor-specific procedures to ensure release of resources within the CCF/SSF so that line, trunk, and other resources are made available for new calls.

NOTE: Depending on the connection view state this should be handled in the physical plane via an ABORT protocol procedure to close the relationship (i.e. close the TCAP transaction) or via the sending of an EntityReleased operation with the relevant cause. This indicates in both cases that any outstanding operations on the corresponding entity (leg or CS) will not be run to completion.

Information available: Once the CCF/SSF has determined an exception condition has occurred, it is assumed the CCF/SSF has information available as when the exception within the PIC occurred.

Exit event: Default handling of the exception condition by the CCF/SSF completed (BCSM transition to O_Null PIC).

6.4.2.2 Terminating BCSM

The terminating half of the BCSM corresponds to that portion of the BCSM associated with the terminating party.
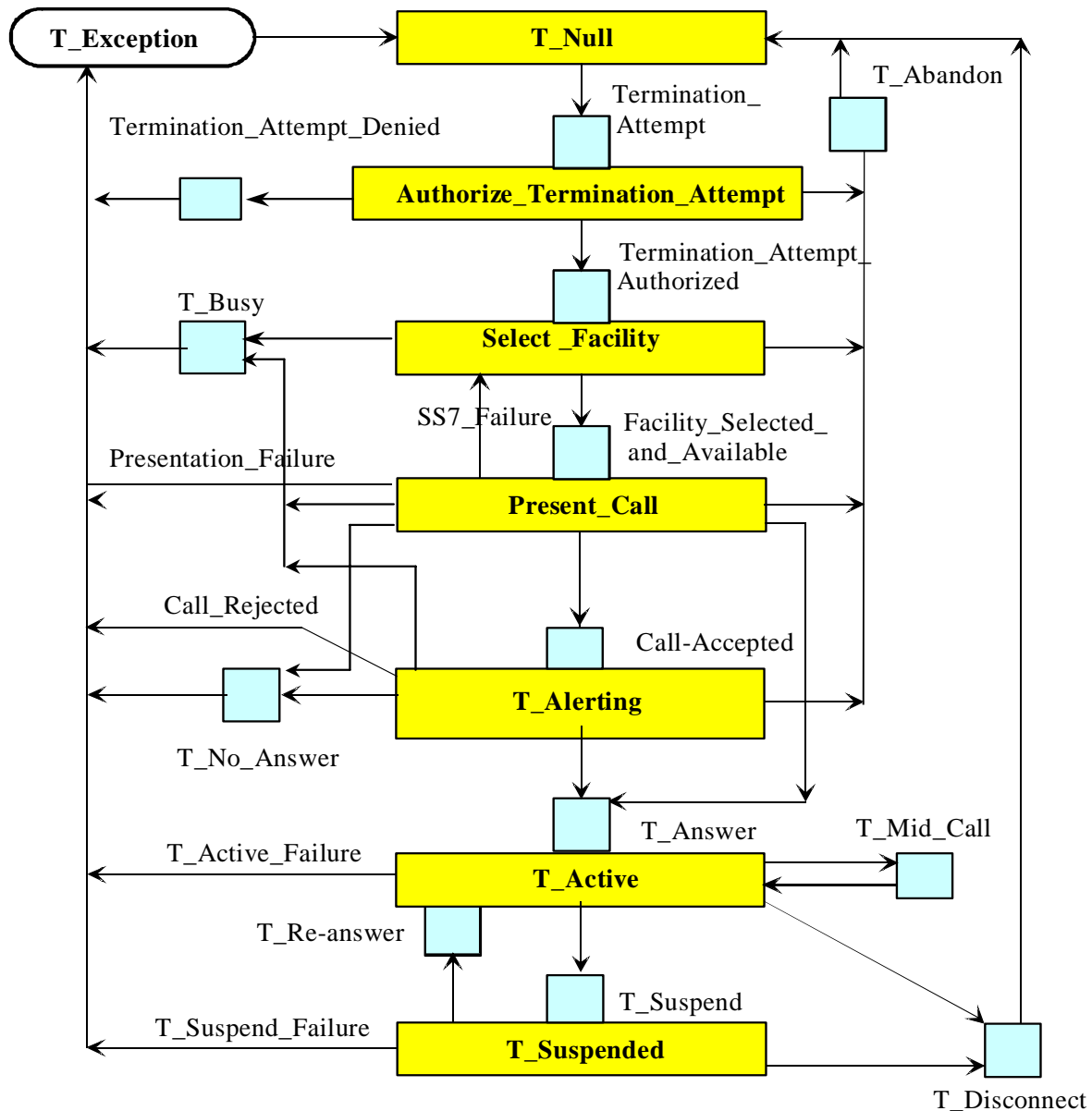
Figure 8: Terminating BCSM for IN CS-3: Basic Set of Transitions

The following information is available at all PICs in the T_BCSM:

- Same information as associated with the originating BCSM is assumed to be available.
  See clause defining the information available at all PICs in the O_BCSM.

- Called Party Class of Service (where locally available e.g. subscriber profile including called party category information).

- Call Reference (where locally assigned in the switch).

- Called Party Terminal Type - see ITU-T Recommendation Q.1290. The SCF uses this to determine the most appropriate form of user-interaction to use (e.g. in-band announcements). This information may only be available at terminating local exchanges.

- Called Party Facility Restriction Level (FRL). The permission level associated with an outgoing facility, e.g. trunk line.

- Called Party BGID (Business Group Identity) - The Basic Business Group or Multiswitch Business Group Identity of the called party, e.g. group-based services.

The above information applies in addition to Signalling Information received from the signalling interface and from the peer O_BCSM. The Signalling Information depends on the signalling interface arrangement applied: DSS.1, DSS.2, analogue user, conventional trunk, SS7 trunk and N-ISUP or B-ISUP, private trunk, H.225, SIP etc. The descriptions for each of the PICs in the terminating half of the BCSM are described below.

> NOTE: See subsequent description on "BCSM Indications for the CS-3 Call Model" for more information concerning PICs.

6.4.2.2.1   T_Null

Entry event: Disconnect and clearing of a previous call (DPs: T_Disconnect or T_Abandon), or default handling of exceptions by CCF/SSF completed.

Function: The Interface (e.g. line/trunk; DSS.1/N-ISDN or DSS.2/B-ISDN) is idled (no call exists, no call reference exists, etc.). Supervision is being provided.

Information available: Once the CCF/SSF has detected the Termination_Attempt event, it is assumed that the CCF/SSF has the information corresponding to the Signalling Information received from the signalling interface obtained from the Setup.Req.Ind Abstract Signalling Primitive available and associated with the terminating portion of the call.
The above information applies in addition to the information available at all PICS in the T_BCSM.

> NOTE 1: information associated with the originating portion of the call as per the Send_Call PIC is assumed to be available. This information is received from the originating call portion, i.e. the peer O_BCSM.

Any information relating to switch-based features that have already been invoked for the call will also be available.

Exit event:

- Indication of incoming call received from originating half BCSM.
  This event causes call processing to move to the Termination_Attempt DP.

- The following exception exit events is applicable to this PIC: T_Abandon.
  If the call encounters T_Abandon during PIC processing, the exception event is not visible because there is no corresponding DP.

Corresponding EN 300 403-1 call state: 0. Null.

> NOTE 2: The T_Abandon event occurs when an indication of call disconnection is received from the originating portion of the call before the call has been answered, i.e. when the calling party disconnects and a Release.Req.Ind Abstract Signalling Primitive is received.

For example, this event can result from one of the following:

- the CCF/SSF receives an on-hook indication from a caller served by a non-ISDN line, following switch-hook flash timing;

- the CCF/SSF receives a call clearing message from a caller served by an ISDN interface;

- the CCF/SSF receives a disconnect indication from a conventional trunk or private facility trunk;

- the CCF/SSF receives a Release Message from an SS7 trunk.

6.4.2.2.2   Authorize_Termination_Attempt

Entry event: Termination_Attempt event (DP: Termination_Attempt).

Function: Verifies the authority to route this call to the terminating access (e.g. DN or trunk group), e.g. check business group restrictions, restricted incoming access to line, or bearer capability compatibility.

Information available: It is assumed that the CCF/SSF has the same information available for the terminating call portion after the Termination_Attempt_Authorized event is detected as it does when the Termination_Attempt event is detected in the T_Null PIC.

- Information as per the T_Null PIC.

- Authorization result - If the CCF/SSF determines that the termination attempt is denied, the cause of the failed authorization is also known.

Exit events:

- Termination_Attempt_Authorized event. This event occurs when the switch has verified the authority to terminate the call to the terminating access. This event causes call processing to move to the Termination_Attempt_Authorized DP.

- The Termination_Attempt_Denied event occurs when the authority to route these call to the terminating user is denied. This causes a BCSM transition to the Termination_Attempt_Denied DP.

- The T_Abandon event occurs when an indication of originating party abandon is received from the originating portion of the call and a Release.Req.Ind Abstract Signalling Primitive is received. This event causes call processing to move to the T_Abandon DP.

6.4.2.2.3   Select Facility

Entry event: Termination_Attempt_Authorized event (DP: Termination_Attempt_Authorized) or an SS7 failure occurs causing a re attempt. The SS7 failure in the Present_Call can be caused by a timer expiry upon sending a continuity check failure.

Function: The busy/idle status of the terminating access is determined.

- For a non-ISDN line, if the line is already involved with an existing call, the line is treated as network-determined user busy.

- For a call terminating to an ISDN interface (on a non-shared DN/CT), network-determined user busy is the detection of one or more of the following conditions:

  - Interface busy: That is, a B-channel is not available for the call.

  - Call-reference busy: There are no idle call reference values available on the terminating DN/CT with which the call can be offered.

    In addition, if the terminating DN is associated with a Multi-Line Hunt Group, busy means that no hunt terminals within the group are available and the queue, if any, is full.

- For conventional trunks, SS7-supported trunks, and private-facility trunks, busy is when all trunks within the selected trunk group are busy.

Information available: When the Facility_Selected and Available event is detected, it is assumed the following information is available and associated with the terminating portion of the call:

- Information as per the T_Null PIC.

- Facility Group - see ITU-T Recommendation Q.1290. For calls routed out of this CCF/SSF, this identifies the Trunk Group (private or public) that has been selected to route the call on. For calls terminating to a non-ISDN line or DSS1 interface within the CCF/SSF, this may identify a particular Multi-line Hunt Group.

- Facility Group Member - see ITU-T Recommendation Q.1290. For calls out of this CCF/SSF, this identifies the trunk (private or public) that has been selected to route the call on. For calls terminating to a non-ISDN line DSS1 interface on the CCF/SSF, this may identify the hunt-terminal within the Multi-line Hunt Group that has been selected for this call.

Exit events:

- Facility_Selected_and_Available event: This event occurs when the terminating access is not busy (i.e. an idle facility [e.g. B-channel, call appearance, or trunk] could be found).
  This event causes call processing to move to the Facility_Selected_and_Available DP.

- A T_Busy event occurs when the terminating access is busy and is detected via a Release.Ind Abstract Signalling Primitive (as defined above). The T_busy event may also be detected as a result of an analogue line being out of order, marked as busy by a customer make-busy key, or as a result of certain maintenance actions.
  (DP: T_Busy).
  The event indication that causes call processing to move to the T_Busy DP depends upon the event reason (cause value) as defined according to table 7.

  After detecting T_Busy, if IN service logic is not needed on the call and no switch-based features apply, an indication of the T_Busy event describing the type of busy (e.g. user or network) is passed to the originating call portion. If a terminating feature acts on the T_Busy event and changes the event (e.g. as in the call Waiting feature), the event is not passed to the Originating BCM.

- The T_Abandon event occurs when an indication of originating party abandon is received from the originating portion of the call via a Release.Req.Ind Abstract Signalling Primitive.
  This event causes call processing to move to the T_Abandon DP.

6.4.2.2.4   Present Call

Entry event: Facility_Selected_and_Available event. (DP: Facility_Selected_and_Available).

Functions: Terminating resource informed of incoming call (e.g. line seizure, EN 300 403-1 Setup message, ISUP IAM message). In the case of an analogue line, ringing is applied.

Information available: When the Call Accepted event is detected, it is assumed the following information is available and associated with the terminating portion of the call:

- Information as per the T_Null PIC.

- Facility Group, Facility Group Member - See description in the Select Facility PIC.

- Information regarding the call connection - This information includes whether the call is end-to-end SS7 or not and whether the originating access is ISDN or non-ISDN.

Exit events:

- Terminating party is being alerted, i.e. a CallProgress(Alerting).Ind Abstract Signalling Primitive is received from the terminating side (e.g. ringing being applied, EN 300 403-1 ALERTING message, ISUP-ACM message).
  This event causes call processing to move to the Call_Accepted DP.

- Call is accepted and answered by terminating party i.e. a Setup.Conf confirmation Abstract Signalling Primitive is received (e.g. terminating party goes off-hook, EN 300 403-1 Connect message received, ISUP answer message received).
  This event causes call processing to move to the T_Answer DP.

- The T_No_Answer event is an IN event or signalling event via a Release.Ind Abstract Signalling Primitive. It can only occur when a T_No_Answer trigger is assigned and detected or when requested by a RequestReportBCSMEvent. It occurs when the terminating party does not answer before the No_Answer timer expires or when e.g. an ISDN user rejects the call with an explicit "no answer" indication". An indication of T_No_Answer event is passed to the originating half of the BCSM. The event indication received that causes call processing to move to the T_No_Answer DP depends upon the event reason (cause value) as defined according to table 7.

- The T_Abandon event occurs when an indication of originating party abandon is received from the originating portion of the call via a Release.Req.Ind Abstract Signalling Primitive.
  This event causes call processing to move to the T_Abandon DP.

- A continuity check failure. (SS7 failure). This event causes call processing to move to the Select Facility PIC.

- Presentation Failure exception event may happen if the call cannot be presented by the T-Busy event. This event causes call processing to move to the T_Exeption.

- The T-Busy event occurs via a Release.Ind Abstract Signalling Primitive from the terminating side if the call cannot be presented (e.g. due to user determined busy, not reachable subscriber condition etc.); this event is notified to the originating call portion (send Call PIC). The event indication that causes call processing to move to the T_Busy DP depends upon the event reason (cause value) as defined according to table 7. Otherwise this event causes call processing to move to the T_Exception.

Corresponding EN 300 403-1 call state: 6. Call present.

6.4.2.2.5   T_Alerting

Entry event: Terminating party is being alerted of incoming call. (DP: Call_Accepted).

Function: An indication is sent to the originating half BCSM that if the terminating party is being alerted. Continued processing of call setup (e.g. ringing, audible ring indication) is taking place. Waiting for the call to be answered by terminating party.

Information available: Once the terminating party is being alerted of the incoming call, it is assumed that the CCF/SSF has the following information available and associated with the terminating portion of the call:

- Information as per the Present_Call PIC.

Exit events:

- Call is accepted and answered by terminating party, i.e. a Setup.Conf Abstract Signalling Primitive is received (e.g. terminating party goes off-hook, EN 300 403-1 CONNect message received, ISUP answer message received).
  This event causes call processing to move to the T_Answer DP.

- The following exception exit events detected via a Release.Ind Abstract Signalling Primitive are applicable to this PIC: call rejected, T_No_Answer, T-Busy and T_Abandon.

- Call rejected exception event may happen when e.g. an ISDN user rejects a call while being alerted.

- The T_No_Answer event from this PIC is the same as the T_No_Answer event defined as an Exit Event from the Present_Call PIC. The T_No_Answer event occurs when the terminating party does not answer before the No_Answer timer expires or when an ISDN user rejects the call with an explicit "no answer" indication". An indication of T_No_Answer event is passed to the originating half of the BCSM. The event indication received that causes call processing to move to the T_No_Answer DP depends upon the event reason (cause value) as defined according to table 7.

- The T-Busy event occurs if the call cannot be presented, due to e.g. ISDN user determined busy, ISUP release message with busy cause, not reachable subscriber etc. This event causes call processing to move to the T_Busy DP and is notified to the originating call portion (Alerting PIC). The events that causes call processing to move to the T_Busy DP depends upon the event reason (cause value) as defined according to table 7.
  Otherwise this event causes call processing to move to the T_Exception.

- The T_Abandon event occurs when an indication of clearing or of originating party abandon is received from the originating portion of the call.
  This event causes call processing to move to the T_Abandon DP.

Comment:

For terminations to SS7 trunk groups, this PIC is entered upon the receipt of an address complete (ACM) message.

Corresponding EN 300 403-1 call states: 7. Call received and 8. Connect request.

6.4.2.2.6   T_Active

Entry events: Call is accepted and answered by terminating party (e.g. terminating party goes off hook, *EN 300 403-1* Connect message received, ISUP answer message received). (DP: T_Answer).

Function: In this PIC several processes may be initiated:

- An indication is sent to the origination half BCSM that the terminating party has accepted and answered the call. Connection established between originating and terminating party. Call supervision is being provided.

- The calling party may be put on hold and returned to the active phase by a service logic.

- In case that a disconnect indication is received via a Release.Req.Ind Abstract Signalling Primitive from the originating side (O-BCSM) or a Release.Ind Abstract Signalling Primitive from the terminating side, this PIC is immediately moved to the T_Disconnect DP without any action.
  As an option, the call can be continued for an appropriate period (retention timer) in order to activate a service or feature request from an originating or terminating party-on the call initiated via a ServiceFeature.Req.Ind/ServiceFeature.Ind Abstract Signalling Primitive.

Information available: Once the call is accepted and answered by the terminating party, it is assumed the following information is available and associated with the terminating portion of the call:

- Information as per T_Alerting.

- Feature Activation Information - A ServiceFeature.Req.Ind Abstract Signalling Primitive is received from the originating party or a ServiceFeature.Ind Abstract Signalling Primitive received from the terminating party e.g. hook-flash, ISDN feature activator, DTMF provided control code.

Exit events:

- A service or a feature request is received from an originating or terminating party, i.e. a ServiceFeature.Req.Ind or a ServiceFeature.Ind Abstract Signalling Primitive (e.g. DTMF provided control code, hook flash, ISDN feature activator, EN 300 403-1 HOLD or RETRIEVE message from originating party only). The detection of this event causes call processing to move to the T_Mid_Call DP.

- A disconnect indication via a NetworkSuspend.Ind Abstract Signalling Primitive (e.g. on-hook,) is received from the terminating party (non-ISDN). This event causes call processing to move to the T_Suspend DP.

- A disconnect indication is received via a Release.Req.Ind Abstract Signalling Primitive from the originating party via the originating half BCSM or a Release.Ind Abstract Signalling Primitive from the terminating party (ISDN). This event causes call processing to move to the T_Disconnect DP.

- A connection failure occurs. (T_Exception).

Comments:

- If originating party abandons call while suspended at T_Answer DP a transition to T_Abandon DP shall occur.

- Disconnect indications and treatment are asymmetrical in the way disconnect timing is applied.

Corresponding EN 300 403-1 call state: 10. Active.

EN 300 403-1 call states corresponding to T_Disconnect: 11. Disconnect request, 12. disconnect indication, and 19. Release request.

6.4.2.2.7   T_Suspended

Entry event: A NetworkSuspend.Ind Abstract Signalling Primitive is received from the outgoing network that the terminating party has disconnected (e.g. on-hook). (DP: T_Suspend).

Function: The physical resources associated with the call remain connected.

- According to the received indication the following applies:

  - A suspend indication, i.e. a NetworkSuspend.Req.Ind Abstract Signalling Primitive may be sent to the originating half BCSM.

  - In case that a disconnect indication via a Release.Ind Abstract Signalling Primitive (e.g. *EN 300 403-1* disconnect message, SS7 release message) is received from the terminating party, this PIC is immediately exited to the T_Disconnect DP without any action.

- In the following cases, the timer is started and the call waits for re-answer request from the terminating party:

1) In case of receiving network initiated suspend message, i.e. a NetworkSuspend.Ind Abstract Signalling Primitive(for an SS7 supported trunk);

2) For an analogue interface, in case of detecting on-hook.

   If re-answer request (e.g. off-hook, SS7 resume message) is received via a NetworkResume.Ind Abstract Signalling Primitive from the terminating party before the timer expires, the originating and terminating parties are reconnected.

   NOTE: Both a Call Resume timer and a Call Retention timer may exist in this PIC. IN implementations may use a single timer for both conditions.

Information available: While in the T_Suspended PIC, it is assumed that the CCF/SSF has the following information available and associated with the terminating call portion.

- Information as per the T_Active PIC.

Exit event:

- The terminating party re-answers or a resume message is received before the timer expires i.e. a NetworkResume.Ind Abstract Signalling Primitive is received; the T_BCSM returns to the T_Active PIC. This event causes call processing to move to the T_Re-answer DP.

- The timer expires. This event causes call processing to move to the T_Disconnect DP.

- A disconnection indication is received via a Release.Ind Abstract Signalling Primitive from the terminating party. This event causes call processing to move to the T_Disconnect DP.

- A disconnection indication is received via a Release.Req.Ind Abstract Signalling Primitive from the originating party. This event causes call processing to move to the T_Disconnect DP.

- An exception event is encountered. (T_Exception).

6.4.2.2.8   T_Exception

Entry event: An exception condition is encountered (as described above for each PIC).

Function: An indication of the exception condition is sent to the originating half BCSM. Default handling of the exception condition is being provided. This includes general actions necessary to ensure no resources remain inappropriately allocated, such as:

- If any relationships exist between the SSF and SCF(s), send an error information to the SCF(s) closing the relationships and indicating that any outstanding call handling instructions will not be run to completion (see note).

- If an SCF previously requested that call parameters be provided at the end of the call (see the CallInformationRequest operation), these should be included in the error information.

- The CCF/SSF should make use of vendor-specific procedures to ensure release of resources within the CCF/SSF so that line, trunk, and other resources are made available for new calls.

NOTE: Depending on the connection view state this should be handled in the physical plane via an ABORT protocol procedure to close the relationship (i.e. close the TCAP transaction) or via the sending of an EntityReleased operation with the relevant cause. This indicates in both cases that any outstanding operations on the corresponding entity (leg or CS) will not be run to completion.

Information available: Once the CCF/SSF has determined an exception condition has occurred, it is assumed the SSF/CF has information available as when the exception within the PIC occurred.

Exit event: Default handling of the exception condition by CCF/SSF completed (BCSM transition to T_Null PIC).

6.4.3   BCSM Resume Points and BCSM Transitions in the IN CS-3 Call Model

6.4.3.1   Originating BCSM: Detailed Set of O_BCSM Transitions
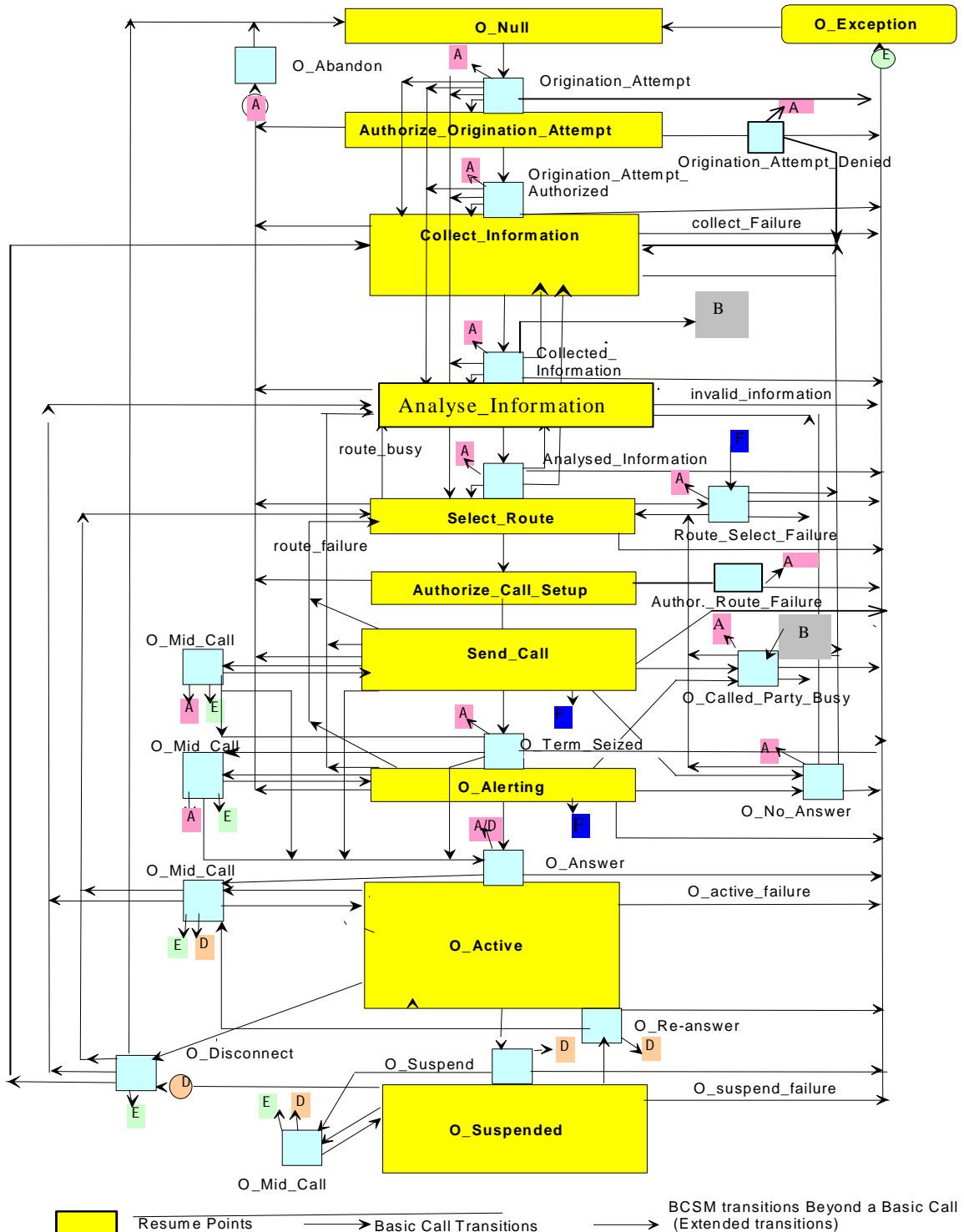
Figure 9: Detailed Set of O_BCSM Transitions

Table 2 together with figure 9 describe the complete set of possible BCSM transitions for the originating call model. The nature of the O_BCSM transitions is given in the third column.

Basic:
Basic transitions refer to normal processing of a basic call, i.e. a two-party call. The basic call process resumes in sequence of PIC and DP (e.g. Continue).

SCF Extended: The basic call process progresses in non-sequence of PIC and DP compared to a basic call due to SCF instruction (e.g. Connect).

CCF Extended: The basic call process progresses due to CCF call control instruction (e.g. Operator signal for call intrusion when call suspended at 'Busy' DP).

Table 2: Complete Set of O_BCSM Transitions for the IN CS-3 Model: DP to DP/PIC_

O_BCSM: Transitions from DP to DP/PIC_		
From	**To**	**Nature of BCSM Transitions**
Origination_Attempt DP	Authorize_Origination_Attempt PIC Collect_Information PIC Analyse_Information PIC Select_Route PIC O_Exception PIC O_Abandon DP.	Basic SCF Extended SCF Extended SCF Extended CCF Extended CCF Extended
Origination_Attempt_Denied DP	Collect_Information PIC O_Abandon DP O_Exception PIC	SCF Extended CCF Extended CCF Basic
Orig._Attempt_Authorized DP	Collect_Information PIC Analyse_Information PIC Select_Route PIC O_Exception PIC O_Abandon DP	Basic SCF Extended SCF Extended CCF Extended CCF Extended
Collected_Information DP	Collect_Information PIC Analyse_Information PIC Select_Route PIC O_Exception PIC O_Abandon DP	SCF Extended Basic SCF Extended CCF Extended CCF Extended
Analyse_Information DP	Collect_Information PIC Analyse_Information PIC Select_Route PIC O_Exception PIC O_Abandon DP O_Called_Party_Busy DP	SCF Extended SCF Extended Basic CCF Extended CCF Extended SCF Extended
Authorize_Route_Failure DP	Analyse_Information PIC Select_Route PIC O_Abandon DP O_Exception PIC	SCF Extended SCF Extended CCF Extended Basic
O_Term_Seized DP	O_Alerting PIC O_Answer DP (note 1) O_Mid_Call DP (note 3) O_Exception PIC O_Abandon DP	Basic CCF Extended CCF Extended CCF Extended CCF Extended
Route_Select_Failure DP	Collect_Information PIC Analyse_Information PIC Select_Route PIC O_Abandon DP O_Exception PIC	SCF Extended SCF Extended SCF Extended CCF Extended Basic
O_Called_Party_Busy DP	Collect_Information PIC Analyse_Information PIC Select_Route PIC O_Abandon DP O_Exception PIC	CCF Extended SCF Extended SCF Extended CCF Extended CCF Basic
O_No_Answer DP	Collect_Information PIC Analyse_Information PIC Select_Route PIC O_Abandon DP O_Exception PIC	CCF Extended SCF Extended SCF Extended CCF Extended Basic

O_BCSM: Transitions from DP to DP/PIC_		
From	**To**	**Nature of BCSM Transitions**
O_Answer DP	O_Active PIC O_Mid_Call DP (note 3) O_Exception PIC O_Disconnect DP (note 2) O_Abandon DP	Basic CCF Extended CCF Extended CCF Extended CCF Extended
O_Suspend DP	O_Suspended PIC O_Disconnect DP O_Mid_Call DP (O_Suspended PIC) O_Exception PIC	Basic CCF Extended CCF Extended CCF Extended
O_Re-Answer DP	O_Active PIC O_Disconnect DP O_Mid_Call_Active DP O_Exception PIC	Basic Extended Extended Extended
O_Mid_Call DP (Send Call PIC)	Send_Call PIC O_Mid_Call DP (Send_Call PIC) O_Term_Seized DP (note 1) O_Answer DP (note 1) O_Exception PIC O_Abandon DP	Basic CCF Extended CCF Extended CCF Extended CCF Extended CCF Extended
O_Mid_Call DP (O_Alerting PIC)	O_Alerting PIC O_Mid_Call DP (O_Alerting PIC) O_Answer DP (note 1) O_Exception PIC O_Abandon DP	Basic CCF Extended CCF Extended CCF Extended CCF Extended
O_Mid_Call DP (O_Active PIC)	O_Active PIC Analyse_Information PIC Select_Route PIC O_Mid_Call DP (O_Active PIC) O_Exception PIC O_Disconnect DP	Basic SCF Extended SCF Extended CCF Extended CCF Extended CCF Extended
O_Mid_Call DP (O_Suspended PIC)	O_Suspended PIC Analyse_Information PIC Select_Route PIC O_Mid_Call DP (O_Suspended PIC) O_Exception PIC	Basic CCF Extended CCF Extended CCF Extended CCF Extended
O_Disconnect DP	O_Null PIC Collect_Information PIC Analyse_Information PIC Select_Route PIC O_Exception PIC O_Disconnect DP	Basic CCF Extended SCF Extended SCF Extended CCF Extended CCF Extended
O_Abandon DP	O_Null PIC	Basic
NOTE 1: The DP reporting ensures that the SCF will receive the events in the correct order. For example in case the FSM for CS is in "Waiting for Instructions" state due to an O_Mid_Call EDP-R (SendCall) being reported and in this state O_Term_Seized and O_Answer events are detected followed by the O_Disconnect event. NOTE 2: Release from called party. NOTE 3: If e.g. a DisconnectLeg (p) or MoveLeg (p) is received while call processing is suspended at DP O_Term_Seized respective O_Answer, the O_BCSM for leg c transits to the O_MidCall DP of the Send_Call PIC respective O_Alerting PIC (not of the O_Alerting PIC respective O_Active PIC as answer has not been sent backward on the c-leg).		

Table 3: Complete Set of O_BCSM Transitions for the IN CS-3 Model: PIC to PIC/DP

O_BCSM: Transitions from PIC to PIC/DP		
From	**To**	**Nature of BCSM Transitions**
O_Null PIC	Origination_Attempt DP	Basic
Authorize_Origination_Attempt PIC	Origination_Attempt_Authorized DP	Basic
	Origination_Attempt_Denied DP	Basic
	O_Abandon DP	Basic
	O_Exception PIC	Basic
Collect_Information PIC	Collected_Information DP	Basic
	O_Abandon DP	Basic
	O_Exception PIC	Basic
Analyse_Information PIC	Analysed_Information DP	Basic
	O_Abandon DP	Basic
	O_Exception PIC	Basic
Select_Route PIC	Analyse_Information PIC	Basic
	Authorize_Call_Setup PIC	Basic
	Route_Select_Failure DP	Basic
	O_Abandon DP	Basic
	O_Exception PIC	Basic
Authorize_Call_Setup PIC	Send_Call PIC	Basic
	Authorize_Route_Failure DP	Basic
	O_Abandon DP	Basic
	O_Exception PIC	Basic
Send_Call PIC	O_Term_Seized DP	Basic
	O_Mid_Call DP (Send_Call PIC)	Basic
	O_Called_Party_Busy DP	Basic
	O_Answer DP	Basic
	O_No_Answer DP	Basic
	Select_Route PIC	Basic
	O_Abandon DP	Basic
	O_Exception PIC	Basic
	Analyse_Information PIC	CCF Extended
O_Alerting PIC	Route_Select_Failure DP	Basic
	O_Mid_Call DP	Basic
	O_Answer DP	Basic
	O_No_Answer DP	Basic
	O_Called_Party_Busy DP	Basic
	O_Abandon DP	Basic
	O_Exception PIC	Basic
	Analyse_Information PIC	CCF Extended
O_Active PIC	O_Midcall DP	Basic
	O_Disconnect DP	Basic
	O_Suspend DP	Basic
	O_Exception PIC	Basic
O_Suspended PIC	O_Re-Answer DP	Basic
	O_Mid_Call DP (O_Suspended PIC)	Basic
	O_Disconnect DP	Basic
	O_Exception PIC	Basic
O_Exception	O_Null PIC	Basic

6.4.3.2 Terminating BCSM: Detailed Set of T_BCSM Transitions
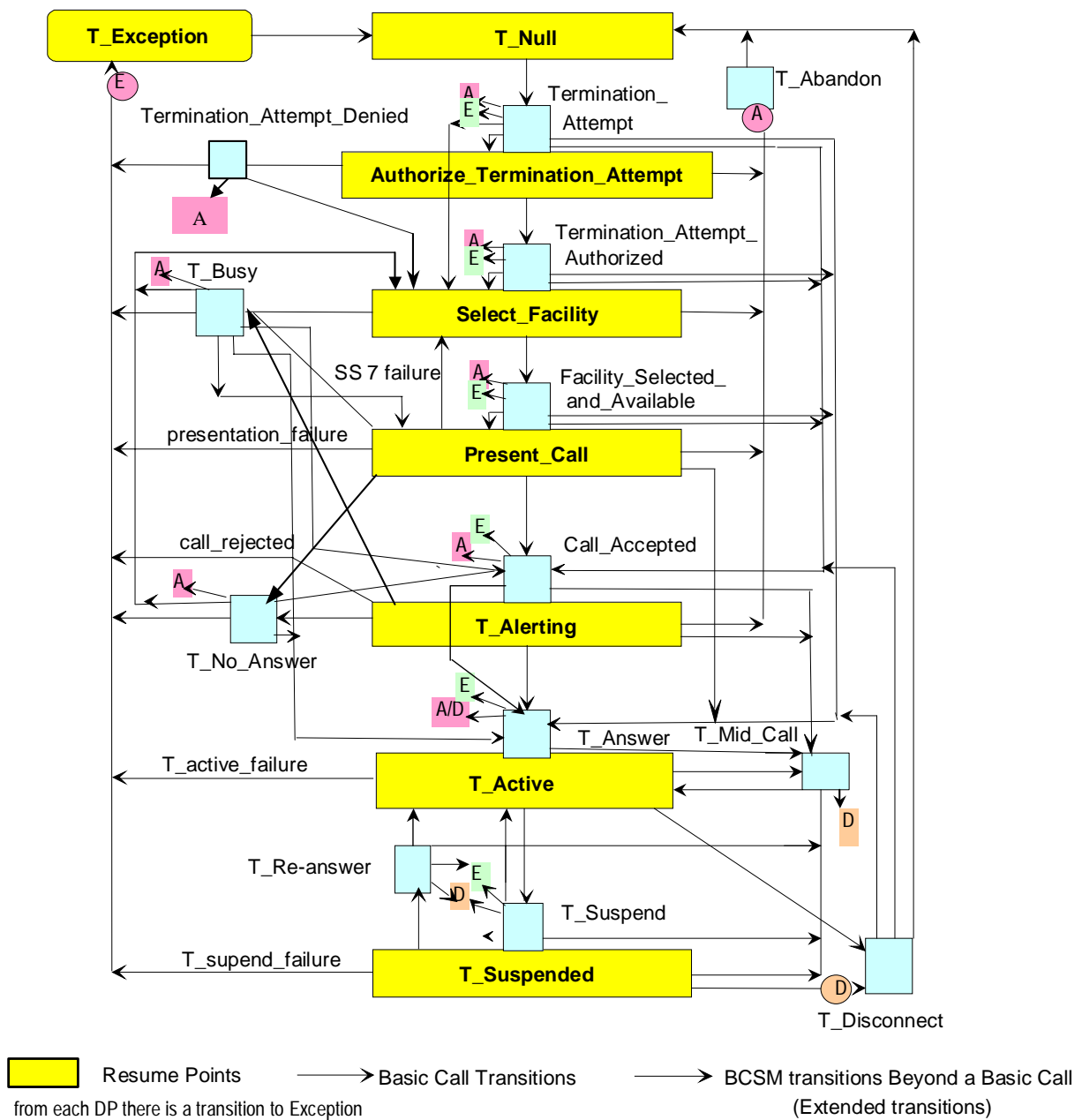
Figure 10: Detailed Set of T_BCSM Transitions

Table 4 together with figure 10 describe the complete set of possible T_BCSM transitions for the terminating call model. The nature of the T_BCSM transitions is given in the third column.

Basic: The basic call process resumes in sequence of DP (e.g. Continue).

Extended: The basic call process progresses in non-sequence of DP compared to a basic call due to SCF instruction PIC or CCF call control instruction.

Table 4: Complete Set of T_BCSM Transitions for the IN CS-3 Model: DP to DP/PIC

From	To	Nature of BCSM Transition
T_BCSM: Transitions from DP to DP/PIC		
Termination_Attempt DP	Authorize_Termination_Attempt PIC	Basic
	Select_Facility PIC	SCF Extended
	Present_Call PIC (note 1)	CCF Extended
	T_Answer DP	CCF Extended
	T_Abandon DP	CCF Extended
	Call Accepted DP	CCF Extended
	T_Exception PIC	CCF Extended
Termination_Attempt_Denied DP	Select_Facility PIC	SCF Extended
	Present_Call PIC (note 1)	SCF Extended
	T_Abandon DP	CCF Extended
	T_Exception PIC	Basic
Terminating_Attempt_Authorized DP	Select_Facility PIC	Basic
	Present_Call PIC (note 1)	SCF Extended
	T_Answer DP	CCF Extended
	T_Abandon DP	CCF Extended
	Call Accepted DP	CCF Extended
	T_Exception PIC	CCF Extended
Facility_Selected_and_Available DP	Present_Call PIC	Basic
	T_Answer DP	CCF Extended
	T_Abandon DP	CCF Extended
	Call Accepted DP	CCF Extended
	T_Exception PIC	CCF Extended
Call_Accepted DP	T_Alerting PIC	Basic
	T_Answer DP (note 2)	CCF Extended
	T_Abandon DP	CCF Extended
	T_Exception PIC	CCF Extended
	T_MidCall DP (note 4)	CCF Extended
T_Busy DP	Select_Facility PIC	SCF Extended
	Present_Call PIC (note 1)	SCF Extended
	T_Answer DP	CCF Extended
	T_Abandon DP	CCF Extended
	Call Accepted DP	CCF Extended
	T_Exception PIC	Basic
T_No_Answer DP	Select_Facility PIC	SCF Extended
	Present_Call PIC (note 1)	SCF Extended
	T_Answer DP	CCF Extended
	Call Accepted DP	CCF Extended
	T_Exception PIC	Basic
T_Answer DP	T_Active PIC	Basic
	T_MidCall DP (note 4)	CCF Extended
	T_Exception PIC	CCF Extended
	T_Disconnect DP (note 3)	CCF Extended
	T_Abandon DP	CCF Extended
T_Suspend DP	T_Suspended PIC	Basic
	T_Disconnect DP	CCF Extended
	T_MidCall DP (T_Active PIC)	CCF Extended
	T_Exception PIC	CCF Extended
T_Re-Answer DP	T_Active PIC	Basic
	T_Disconnect DP	CCF Extended
	T_MidCall DP (T_Active PIC)	CCF Extended
	T_Exception PIC	CCF Extended
T_Midcall DP	T_Active PIC	Basic
	T_Disconnect DP	CCF Extended
	T_Exception PIC	CCF Extended
	T_Midcall DP	CCF Extended

T_BCSM: Transitions from DP to DP/PIC		
From	**To**	**Nature of BCSM Transition**
T_Disconnect DP	T_Null PIC	Basic
	T_Answer DP	CCF Extended
	Call_Accepted DP	CCF Extended
	T_Disconnect DP	CCF Extended
	Present_Call PIC (note 1)	SCF Extended
T_Abandon DP	T_Null PIC	Basic
NOTE 1: When a Connect operation is received the T_BCSM may be suspended in one of the indicated DPs. The T_BCSM moves to the Present_Call PIC.		
NOTE 2: The DP reporting ensures that the SCF will receive the events in the correct order. For example in case the **FSM for CS** is in "Waiting for Instructions" state due to CallAccepted EDP-R being reported and in this state an T-Answer event is detected followed by the T_Disconnect event.		
NOTE 3: Release from called party.		
NOTE 4: If e.g. a DisconnectLeg (c) or MoveLeg (c) is received while call processing is suspended at DP Call_Accepted respective T_Answer, the T_BCSM for leg p transits to a T_MidCall DP "Wait state" of the Present_Call PIC respective T_Alerting PIC (not of the T-Alerting PIC respective T_Active PIC as answer has not been sent backward on the p-leg). Notice that this Mid_Call DP is internal to the T_BCSM model (no Mid_Call DP reported to the SCF) and therefore not shown in the T_BCSM.		

Table 5: Complete Set of T_BCSM Transitions for the IN CS-3 Model: PIC to PIC/DP

T_BCSM: Transitions from PIC to PIC/DP		
From	**To**	**Nature of BCSM Transition**
T_Null PIC	Termination_Attempt DP	Basic
Authorize_Termination_Attempt PIC	Term_Attempt_Authorized DP	Basic
	Termination_Attempt_Denied DP	Basic
	T_Abandon DP	Basic
	T_Exception PIC	Basic
Select_Facility PIC	Facility_Selected_and_Available DP	Basic
	T_Busy DP	Basic
	T_Abandon DP	Basic
Present_Call PIC	T_No_Answer DP	Basic
	T_Answer DP	Basic
	Call_Accepted DP	Basic
	T_Alerting PIC	Basic
	Select_Facility PIC	Basic
	T_Abandon DP	Basic
	T_Busy DP	Basic
	T_Exception PIC	Basic
T_Alerting PIC	T_Answer DP	Basic
	T_No_Answer DP	Basic
	T_Abandon DP	Basic
	T_MidCall DP (note 1)	SCF Extended
	T_Busy DP	Basic
	T_Exception PIC	Basic
T_Active PIC	T_Midcall DP	Basic
	T_Disconnect DP	Basic
	T_Suspend DP	Basic
	T_Exception PIC	Basic
T_Suspended PIC	T_Re-Answer DP	Basic
	T_Disconnect DP	Basic
	T_MidCall DP (T_Active PIC, see note)	SCF Extended
	T_Exception PIC	Basic
T_Exception	T_Null PIC	Basic
NOTE: Notice that this Mid_Call DP may be internal to the T_BCSM model (i.e. no Mid_Call DP reported to the SCF due to a CPH operation) and therefore it may not shown in the T_BCSM.		

6.4.4   BCSM indications for the CS-3 Call Model

6.4.4.1   User - O_BCSM Access Signalling Indications (Category 1)

Definition:

These Indications include the representation of the network's perception of possible actions taken by the calling party as well as the calling party's perception of actions taken by the network.
The Indications are between a user (i.e. calling party) and a local exchange that is originating a call. They include the definition of how actions by the user (originating call model) affect the originating call model (user). These Indications are derived from Access Signalling (e.g. DSS1, analogue) as well as any other information that is available.

Figure 11 illustrates these indications.

List of Indications:

1) An Indication is sent from User to O_BCSM to initiate call establishment (e.g. Setup.Ind Abstract Signalling Primitive mapped from a SETUP message).

2) An Indication is sent from O_BCSM to User that network is unable to initiate call (e.g. Release.Req Abstract Signalling Primitive mapped to RELEASE message).

3) An Indication is sent from O_BCSM to User acknowledging the call initiation Indication (e.g. CallProgress(Progress).Req Abstract Signalling Primitive mapped to a SETUP_ACKNOWLEDGE).

4) The User sends call (dialling) information to the O_BCSM (e.g. SubsequentAddress Abstract Signalling Primitive mapped from INFORMATION).

5) An Indication is sent from O_BCSM to the User to terminate the sending of call information (e.g. CallProgress(Progress).Req Abstract Signalling Primitive mapped to CALL_PROCEEDING).

6) An Indication is sent from the User to the O_BCSM upon completion of call information (e.g. SubsequentAddress Abstract Signalling Primitive mapped from SAM).

7) User is informed that call has been routed to another environment of network (e.g. CallProgress(Progress).Req Abstract Signalling Primitive mapped to a PROGRESS).

8) An Indication is sent from the O_BCSM to the User when the called party is being alerted (e.g. CallProgress(Alert).Req Abstract Signalling Primitive mapped to a ALERTING).

9) An Indication is sent from the O_BCSM to the User when the call is accepted. (e.g. Setup.Resp Abstract Signalling Primitive is mapped to CONNECT).

10) The User acknowledges that the call is accepted.

11) The O_BCSM sends an Indication to the User that the called party is unable to accept the call, due to busy condition.

12) The O_BCSM sends an Indication to the User since the called party is unable to accept the call, due to no answer condition.

13) An Indication is received by the O_BCSM from the User to end the call.

14) The O_BCSM indicates to the User that the call is being disconnected.

15) The User acknowledges to the O_BCSM that the call is being disconnected.

16) An Indication is sent to the user when the connection towards the Called Party is suspended.

17) An Indication is sent to the user when the connection towards the Called Party is reconnected.

(1)

O_Null

O_Exception

O_Abandon DP

(2) Origination Attempt DP

Auth_Origination_Attempt

(3)

Origination_Attempt_Denied DP

Origination_Attempt_Authorized DP

(4) Collect Information

Collect_Failure

Collected Information DP

(5) Analyse Information

(6) Invalid_Information

Route_Busy Analysed_Information DP

Select Route

Route_Select_Failure DP

Route_Failure

Auth_Call_Setup

Authorize_Route_Failure DP

(7) O_Called_Party_Busy DP

O_Mid_Call DP

Send_Call

(11)

O_Term_Seized DP

(8) O_Mid_Call DP O_No_Answer DP

O_Alerting

(12)

(9) O_Answer DP

(10)

O_Mid_Call DP O_Active

(13) o_active_failure

(17)

O_Re_Answer DP

(16) O_Suspend DP

(14)

O_Suspended

(15) o_suspend_failure 7

O_Disconnect DP

O_Mid_Call DP

→ BCSM Transition

⇢ Category 1 Indication

NOTE: Indications which are shown as terminating on a DP are received by the switch and are not part of the IN
DP Processing.

Figure 11: Access Signalling Indications for the CS-3 BCSM Category 1 (User - O_BCSM)

6.4.4.2   T_BCSM - User Access Signalling Indications (Category 2)

Definition:

These Indications include the representation of the network's perception of possible actions taken by the called party as well as the called party's perception of actions taken by the network. The Indications are between a local exchange that is terminating a call and a user (i.e. called party). They include the definition of how actions by the terminating call model (user) affect the user (terminating call model). These Indications are derived from Access Signalling (e.g. DSS1, analogue) as well as any other information that is available.

Figure 12 illustrates these Indications.

List of Indications:

1) An Indication is sent from T_BCSM to the User to terminate the call to an idle facility (e.g. Setup.Req Abstract Signalling Primitive mapped to SETUP).

2) An Indication is sent from User to T_BCSM indicating that the User cannot accept the call (e.g. Release.Req Abstract Signalling Primitive mapped to RELEASE message).

3) An Indication is sent from the User to the T_BCSM when the User determines compatibility with all call characteristics (e.g. CallProgress(Progress). Ind Abstract Signalling Primitive mapped from a SETUP_ACKNOWLEDGE).

4) The T_BCSM sends any remaining call information to the User (e.g. INFORMATION).

5) An Indication is sent from the T_BCSM to the User upon the sending of sufficient call information.

6) An Indication is sent from the User to the T_BCSM upon receipt of sufficient call information (e.g. CallProgress(Progress).Ind Abstract Signalling Primitive SubsequentAddress Abstract Signalling Primitive mapped from CALL_PROCEEDING).

7) User sends an Indication to the T_BCSM that alerting is taking place (e.g. CallProgress(Alert).Ind Abstract Signalling Primitive mapped from ALERTING).

8) An Indication is sent from the User to the T_BCSM upon acceptance of the incoming call (e.g. Setup Conf. Abstract Signalling Primitive mapped from CONNECT).

9) An Indication is sent from the T_BCSM to the User acknowledging that the call can now be connected.

10) An Indication is sent from the User to the T_BCSM that the User suspends the call.

11) An Indication is sent from the User to the T_BCSM that the User resumes the call.

12) The T_BCSM sends an Indication to the User indicating that the calling party has gone on-hook.

13) An Indication is received by the T_BCSM from the User to end the call.

14) The T_BCSM indicates to the User that the call is being disconnected.

15) The User acknowledges to the T_BCSM that the call is being disconnected.
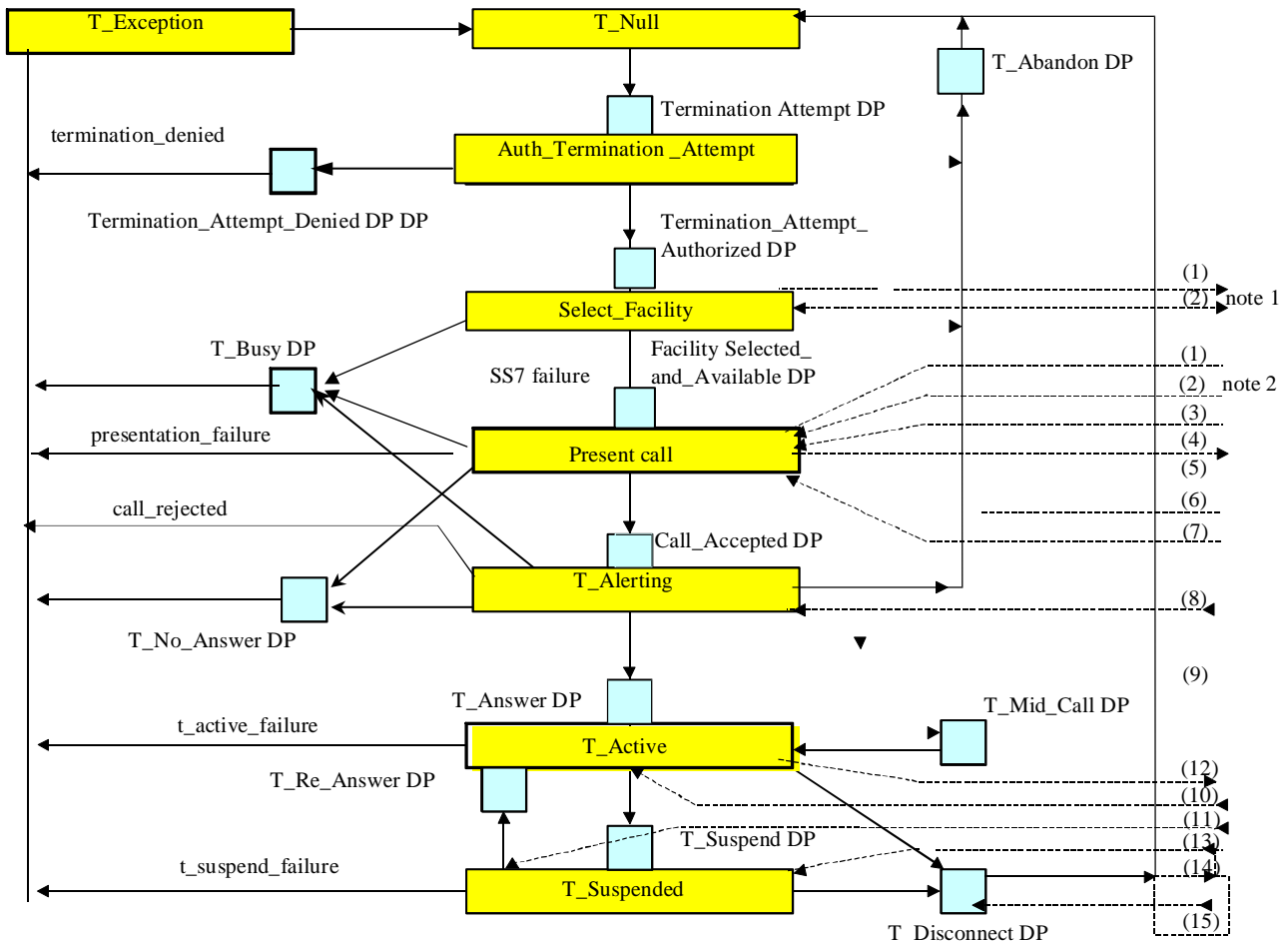
BCSM Transitions

Category 2 Indications

NOTE 1: Indications 1 and 2 at Select_Facility for case of switch-CPE B channel negotiation.
NOTE 2: Indications 1 and 2 at Present_Call for case of no B channel negotiation.
NOTE 3: Indications which are shown as terminating on a DP are received by the switch and are not part of the IN DP processing.

Figure 12: Access Signalling for the CS-3 BCSM Category 2 (T_BCSM-User)

6.4.4.3 Intra Local Exchange BCSM Indications (Category 3)

Figure 13 illustrates the communication between two call segments in the CCF/SSF for a basic two-party call, as described in the CCF/SSF Model. It shows the indications that flow between the originating BCSM and terminating BCSM. All possible indications are shown, except for any which may occur at the O-Exception and the T-Exception PICs. Note that these indications are not intended to be mapped to explicit information flows.

1) Initiate T_BCSM after the authority to place the call has been verified and a usable route has been identified. The O_BCSM is currently in the Send_Call PIC. The originating Basic Call Manager has sent the call attempt to the terminating Basic Call Manager for further processing, i.e. the Setup.Req.Ind Abstract Signalling Primitive is sent on the IBI from the O-BCSM to the T_BCSM.

2) For SS7-supported trunks, if the received IAM indicates a continuity check is required and the resultant continuity check is successful, then an Indication is sent from the O_BCSM to the T_BCSM (causes T_Null PIC to Termination_Attempt DP BCSM transition in the T_BCSM).

3) An Indication is sent from the T_BCSM to O_BCSM that the terminating line or trunk is busy, i.e. the Release.Req.Ind Abstract Signalling Primitive is sent on the IBI from the T-BCSM to the O_BCSM (causes Send_Call PIC to O_Called_Party_Busy BCSM transition in O_BCSM).

4) An Indication is sent from the T_BCSM to O_BCSM that the terminating line or trunk is busy, i.e. the Release.Req.Ind Abstract Signalling Primitive is sent on the IBI from the T-BCSM to the O_BCSM (causes O_Alerting PIC to O_Called_Party_Busy DP BCSM transition in O_BCSM).

5) An Indication is sent from the T_BCSM to O_BCSM that the call cannot be presented, i.e. the Release.Req.Ind Abstract Signalling Primitive is sent on the IBI from the T-BCSM to the O_BCSM (causes Send_Call PIC to Select_Route PIC, O_Called_Party_Busy DP, or O_No_Answer DP).

6) An Indication is sent from the T_BCSM to the O_BCSM that an ISDN capable Called Party has signalled call acceptance with immediate BCSM transition to an answered condition, i.e. the Setup.Resp.Conf Abstract Signalling Primitive is sent on the IBI from the T-BCSM to the O_BCSM (causes Send_Call PIC to O_Answer DP BCSM transition in O_BCSM).

7) An Indication is sent from T_BCSM to O_BCSM that Called Party is being alerted i.e. the CallProgress(Alert).Req.Ind Abstract Signalling Primitive is sent on the IBI from the T-BCSM to the O_BCSM (causes O_BCSM to transit from Send_Call PIC O_Alerting PIC and prepare to send ring Indication to the Calling Party).

8) An Indication is sent from T_BCSM to O_BCSM that Called Party has rejected the call, i.e. the Release.Req.Ind Abstract Signalling Primitive is sent on the IBI from the T-BCSM to the O_BCSM (this is indicated to the O_BCSM with a busy cause and causes O_BCSM to transit from O_Alerting PIC to Select_Route PIC or O_Called_Party_Busy DP).

9) An Indication is sent from T_BCSM to O_BCSM that Called Party has not answered within a specified time period, i.e. the Release.Req.Ind Abstract Signalling Primitive is sent on the IBI from the T_BCSM to the O_BCSM (causes O_Alerting PIC to O_No_Answer DP BCSM transition in O_BCSM).

10) An Indication is sent from the T_BCSM to the O_BCSM that called party has not answered within a specified time period, i.e. the Release.Req.Ind Abstract Signalling Primitive is sent on the IBI from the T_BCSM to the O_BCSM (causes Send_Call PIC to O_No_Answer DP BCSM transition in O_BCSM).

11) An Indication is sent from T_BCSM to O_BCSM that Called Party has accepted and answered the call attempt, i.e. the Setup.Resp.Conf Abstract Signalling Primitive is sent on the IBI from the T_BCSM to the O_BCSM (causes O_Alerting PIC to O_Answer DP BCSM transition in O_BCSM).

12) An Indication is sent from the T_BCSM to the O_BCSM that the called party has accepted and answered the call attempt, i.e. the Setup.Resp.Conf Abstract Signalling Primitive is sent on the IBI from the T-BCSM to the O_BCSM (causes Send_Call PIC to O_Answer DP BCSM transition in O_BCSM).

13) An Indication is sent from T_BCSM to O_BCSM that Called Party has disconnected, i.e. the NetworkSuspend.Req.Ind Abstract Signalling Primitive is sent on the IBI from the T_BCSM to the O_BCSM (e.g. on-hook), (causes O_Active PIC to O_Suspend DP BCSM transition in O_BCSM).

14) An Indication is sent from T_BCSM to O_BCSM that Called Party re-answers is received before the timer expires, i.e. the NetworkResume.Req.Ind Abstract Signalling Primitive is sent on the IBI from the T-BCSM to the O_BCSM (causes O_Suspended PIC to O_Re_Answer DP BCSM transition in O_BCSM). Note that the name and function of this timer is FFS.

15) An Indication is sent from O_BCSM to T_BCSM that Calling Party has disconnected, i.e. the Release.Req.Ind Abstract Signalling Primitive is sent on the IBI from the O_BCSM to the T_BCSM, while T_BCSM was in T_Active PIC (causes T_Active PIC to T_Disconnect DP BCSM transition in T_BCSM).

16) An Indication is sent from O_BCSM to T_BCSM that Calling Party has disconnected,, i.e. the Release.Req.Ind Abstract Signalling Primitive is sent on the IBI from the O_BCSM to the T_BCSM, while T_BCSM was in T_Suspended PIC (causes T_Suspended PIC to T_Disconnect DP BCSM transition in T_BCSM).

17) An Indication is sent from T_BCSM to O_BCSM that Called Party has disconnected, i.e. the Release.Req.Ind Abstract Signalling Primitive is sent on the IBI from the T_BCSM to the O_BCSM, (causes O_Suspended PIC to O_Disconnect DP BCSM transition in O_BCSM).

18) An Indication is sent from the T_BCSM (T_Disconnect DP) to O_BCSM that the called party has disconnected, i.e. the Release.Req.Ind Abstract Signalling Primitive is sent on the IBI from the T_BCSM to the O_BCSM, (causes O_Active PIC to O_Disconnect DP BCSM transition in O_BCSM).

19) An Indication is sent from O_BCSM to T_BCSM that Calling Party has abandoned, i.e. the Release.Req.Ind Abstract Signalling Primitive is sent on the IBI from the O_BCSM to the T_BCSM, (causes Authorize_Termination_Attempt PIC, Select_Facility PIC, Present_Call PIC or T_Alerting PIC to T_Abandoned DP BCSM transition in T_BCSM).

NOTE: Indications (15) and (17) are mutually exclusive:

- these indications are for intra-switch;

- the indications do not explicitly include the modelling of SRFs;

- indications which are preceded by a DP may be affected depending on whether the DP is active and the SCF response.
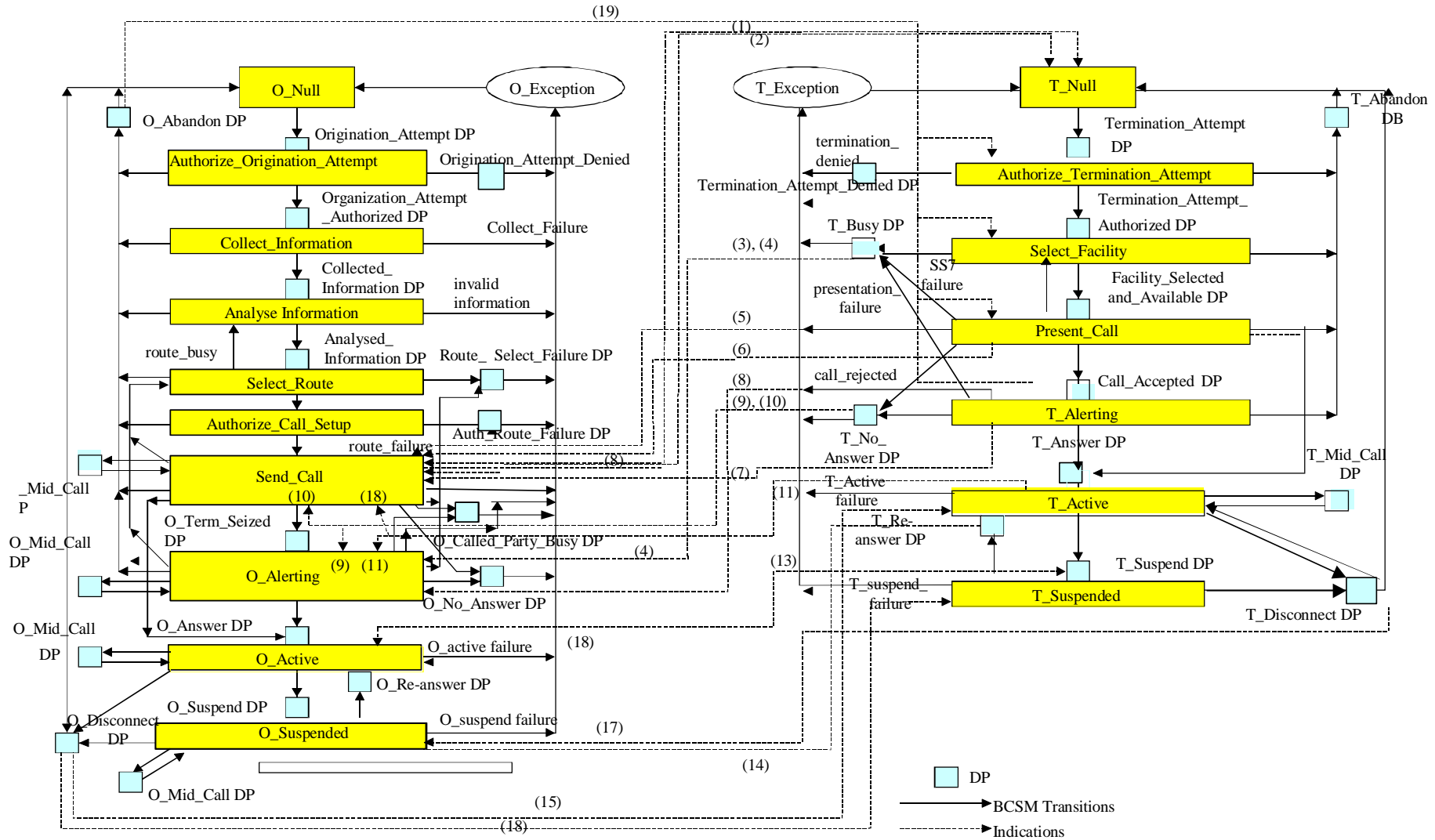
Figure 13: Intra Local Exchange BCSM Indications (Category 3)

6.4.5   Mapping from Cause to DP

A normative mapping between signalling release cause values and DPs is defined by Mapping Tables, i.e. the table covers only cause values received by the basic call signalling.
The tables are based on the Call States. In some cases the mapping of a particular cause value to a DP depends on the PIC in which the release message is received. This is indicated where appropriate by notes.

This Mapping from Cause value to DP as indicated in the tables below for O_BCSM and respectiveT_BCSM is normative, except where the table indicates that receipt of the cause value leads to the O/T-Exception PIC. In the latter case, network operators may decide to map some of these cause values to a specific DP.

6.4.5.1   O_BCSM: Mapping Table from Cause to DP

A flexible mapping is provided as the DP_Route_Select Failure is used as a possible transition for e.g. the PICs Send_Call and O_Alerting. The transitions are done through an internal transition in the O_BCSM.

The following mapping of O_BCSM PICs is applied:

For ease of reference, the PICs have been categorized as follows:

- Category: **Originating Call setup**:
  PICs: Authorize_Origination_Attempt, Collect_Information, Analyse_Information, Select_Route, Authorize_Call_Setup.

- Category: **Originating Stable Call**:
  PICs: Send-Call, O_Alerting, O_Active.

- Category: **Originating Call Clearing**:
  PICs: O_Suspended.

Table 6: O_BCSM: Mapping Cause value to DP

No.	Reason (see ITU-T Recommendation Q.850)	Originating CallSetup	Originating Stable Call		Originating CallClearing
			Send_Call, O-Alerting (see note)	O-Active	
1	Unallocated (unassigned) number	Route_Select_Failure	Route_Select_Failure	Exception	Exception
2	No route to specified transit network	Route_Select_Failure	Route_Select_Failure	Exception	Exception
3	No route to destination	Route_Select_Failure	Route_Select_Failure	Exception	Exception
4	Send special information tone	Route_Select_Failure	Route_Select_Failure	Exception	Exception
5	Misdialled trunk prefix	Route_Select_Failure	Route_Select_Failure	Exception	Exception
6	Channel unacceptable (Q931 only)	Exception	Exception	Exception	Exception
7	Call awarded and being delivered in an established channel (Q931 only)	Exception	Exception	Exception	Exception
8	Pre-emption	Route_Select_Failure	Route_Select_Failure	Exception	Exception
9	Pre-emption – circuit reserved for reuse	Route_Select_Failure	Route_Select_Failure	Exception	Exception
14	Ported subscriber	Route_Select_Failure	Route_Select_Failure	Exception	Exception
16	Normal call clearing	Route_Select_Failure (backward direction) O-Abandon (forward direction)	Route_Select_Failure (backward direction) O_Abandon (forward direction)	O_Disconnect	O_Disconnect

No.	Reason (see ITU-T Recommendation Q.850)	Originating CallSetup	Originating Stable Call		Originating CallClearing
			Send_Call, O-Alerting (see note)	O-Active	
17	User busy	Exception	O_Called_Party_Busy	Exception	Exception
18	No user responding	Exception	O_No_Answer	Exception	Exception
19	No answer from user (user alerted)	Exception	O_No_Answer	Exception	Exception
20	Subscriber absent	Exception	O_Called_Party_Busy	Exception	Exception
21	Call rejected	Route_Select_Failure	Route_Select_Failure	Exception	Exception
22	Number changed	Route_Select_Failure	Route_Select_Failure	Exception	Exception
26	Non-selected user clearing (Q931 only)	Exception	Exception	Exception	Exception
27	Destination out of order	Route_Select_Failure	Route_Select_Failure	Exception	Exception
28	Invalid number format (address incomplete)	Route_Select_Failure	Route_Select_Failure	Exception	Exception
29	Facility rejected	Route_Select_Failure	Route_Select_Failure	Exception	Exception
30	Response to STATUS ENQUIRY (Q931 only)	Exception	Exception	Exception	Exception
31	Normal, unspecified	Route_Select_Failure (backward direction) O_Abandon (forward direction)	Route_Select_Failure (backward direction) O_Abandon (forward direction)	O_Disconnect	O_Disconnect
34	No circuit/channel available	Exception	O_Called_Party_Busy	Exception	Exception
38	Network out of order	Route_Select_Failure	Route_Select_Failure	Exception	Exception
39	Permanent frame mode connection out of service (Q931 only)	Exception	Exception	Exception	Exception
40	Permanent frame mode connection operational (Q931 only)	Exception	Exception	Exception	Exception
41	Temporary failure	Route_Select_Failure	Route_Select_Failure	Exception	Exception
42	Switching equipment congestion	Route_Select_Failure	Route_Select_Failure	Exception	Exception
43	Access information discarded	Route_Select_Failure	Route_Select_Failure	O_Disconnect	Disconnect
44	Requested circuit/channel not available	Exception	O_Called_Party_Busy	Exception	Exception
46	Precedence call blocked	Route_Select_Failure	Route_Select_Failure	Exception	Exception
47	Resource unavailable, unspecified	Route_Select_Failure	Route_Select_Failure	Exception	Exception
49	Quality of service unavailable	Route_Select_Failure	Route_Select_Failure	Exception	Exception
50	Requested facility not subscribed	Route_Select_Failure	Route_Select_Failure	Exception	Exception
53	Outgoing calls barred within CUG	Route_Select_Failure	Route_Select_Failure	Exception	Exception
55	Incoming calls barred within CUG	Route_Select_Failure	Route_Select_Failure	Exception	Exception
57	Bearer capability not authorized	Route_Select_Failure	Route_Select_Failure	Exception	Exception
58	Bearer capability not presently available	Route_Select_Failure	Route_Select_Failure	Exception	Exception
62	Inconsistency in designated outgoing access information and subscriber class	Exception	Exception	Exception	Exception
63	Service or option not available, unspecified	Route_Select_Failure	Route_Select_Failure	Exception	Exception
65	Bearer capability not implemented	Route_Select_Failure	Route_Select_Failure	Exception	Exception

No.	Reason (see ITU-T Recommendation Q.850)	Originating CallSetup	Originating Stable Call		Originating CallClearing
			Send_Call, O-Alerting (see note)	O-Active	
66	Channel type not implemented (Q931 only)	Exception	Exception	Exception	Exception
69	Requested facility not implemented	Exception	Exception	Exception	Exception
70	Only restricted digital information bearer capability is available	Route_Select_Failure	Route_Select_Failure	Exception	Exception
79	Service or option not implemented, unspecified	Route_Select_Failure	Route_Select_Failure	Exception	Exception
81	Invalid call reference value (Q931 only)	Exception	Exception	Exception	Exception
82	Identified channel does not exist (Q931 only)	Exception	Exception	Exception	Exception
83	A suspended call exists, but this call identity does not (Q931 only)	Exception	Exception	Exception	Exception
84	Call identity in use (Q931 only)	Exception	Exception	Exception	Exception
85	No call suspended (Q931 only)	Exception	Exception	Exception	Exception
86	Call having the requested call identity has been cleared (Q931 only)	Exception	Exception	Exception	Exception
87	User not member of CUG	Route_Select_Failure	Route_Select_Failure	Exception	Exception
88	Incompatible destination	Route_Select_Failure	Route_Select_Failure	Exception	Exception
90	Non-existent CUG	Route_Select_Failure	Route_Select_Failure	Exception	Exception
91	Invalid transit network selection	Route_Select_Failure	Route_Select_Failure	Exception	Exception
95	Invalid message, unspecified	Exception	Exception	Exception	Exception
96	Mandatory information element is missing	Exception	Exception	Exception	Exception
97	Message type non-existent or not implemented	Exception	Exception	Exception	Exception
98	Message not compatible with call state or message type non-existent or not implemented	Exception	Exception	Exception	Exception
99	Information element/parameter non-existent or not implemented	Exception	Exception	Exception	Exception
100	Invalid information element contents	Exception	Exception	Exception	Exception
102	Recovery on timer expire	Exception	Exception	Exception	Exception
103	Parameter non-existent or not implemented, passed on	Exception	Exception	Exception	Exception
110	Message with unrecognized parameter, discarded	Exception	Exception	Exception	Exception
111	Protocol error, unspecified	Exception	Exception	Exception	Exception
127	Interworking, unspecified	Route_Select_Failure	Route_Select_Failure	Exception	Exception
NOTE: A transition to Route_Select_Failure DP occurs directly due to the receipt of a route failure event from the called destination.					

6.4.5.2   T_BCSM: Mapping Table from Cause to DP

The transitions are done through an internal transition in the BCSM.

The following mapping of T_BCSM PICs is applied:

For ease of reference, the PICs have been categorized as follows:

- Category: Terminating Call setup:

  PICs: Authorize_termination_Attempt, Select_Facility, Present_Call.

- Category: Terminating Stable Call:

  PICs: T_Alerting., T_Active.

- Category: Terminating Call Clearing:

  PICs: T_Suspended.

Table 7: T_BCSM: Mapping Cause value to DP

No.	Reason (see ITU-T Recommendation Q.850)	Terminating CallSetup	Terminating Stable Call		Terminating Call Clearing
			T-Alerting	**T-Active**	
1	Unallocated (unassigned) number	Exception	Exception	Exception	Exception
2	No route to specified transit network	Exception	Exception	Exception	Exception
3	No route to destination	Exception	Exception	Exception	Exception
4	Send special information tone	Exception	Exception	Exception	Exception
5	Misdialled trunk prefix	Exception	Exception	Exception	Exception
6	Channel unacceptable (Q931 only)	Exception	Exception	Exception	Exception
7	Call awarded and being delivered in an established channel (Q931 only)	Exception	Exception	Exception	Exception
8	Pre-emption	Exception	Exception	Exception	Exception
9	Pre-emption – circuit reserved for reuse	Exception	Exception	Exception	Exception
14	Ported subscriber	Exception	Exception	Exception	Exception
16	Normal call clearing	Exception (backward direction) T_Abandon (forward direction)	Exception (backward direction) T_Abandon (forward direction)	T_Disconnect	T_Disconnect
17	User busy	T_Busy (see note 2)	T_Busy	Exception	Exception
18	No user responding	T_No_Answer (see note 1)	Exception	Exception	Exception
19	No answer from user (user alerted)	Exception	T_No_ Answer	Exception	Exception
20	Subscriber absent	T_Busy (see note 2)	T_Busy	Exception	Exception
21	Call rejected	Exception	Exception	Exception	Exception
22	Number changed	Exception	Exception	Exception	Exception
26	Non-selected user clearing (Q931 only)	Exception	Exception	Exception	Exception
27	Destination out of order	Exception	Exception	Exception	Exception
28	Invalid number format (address incomplete)	Exception	Exception	Exception	Exception
29	Facility rejected	Exception	Exception	Exception	Exception

No.	Reason (see ITU-T Recommendation Q.850)	Terminating CallSetup	Terminating Stable Call		Terminating Call Clearing
			T-Alerting	T-Active	
30	Response to STATUS ENQUIRY (Q931 only)	Exception	Exception	Exception	Exception
31	Normal, unspecified	Exception (backward direction) T_Abandon (forward direction)	Exception (backward direction) T-Abandon (forward direction)	T_Disconnect	T_Disconnect
34	No circuit/channel available	T_Busy (see note 2)	T_Busy	Exception	Exception
38	Network out of order	Exception	Exception	Exception	Exception
39	Permanent frame mode connection out of service (Q931 only)	Exception	Exception	Exception	Exception
40	Permanent frame mode connection operational (Q931 only)	Exception	Exception	Exception	Exception
41	Temporary failure	Exception	Exception	Exception	Exception
42	Switching equipment congestion	Exception	Exception	Exception	Exception
43	Access information discarded	Exception	Exception	T_Disconnect	T_Disconnect
44	Requested circuit/channel not available	T_Busy (see note 2)	T_Busy	Exception	Exception
46	Precedence call blocked	Exception	Exception	Exception	Exception
47	Resource unavailable, unspecified	Exception	Exception	Exception	Exception
49	Quality of service unavailable	Exception	Exception	Exception	Exception
50	Requested facility not subscribed	Exception	Exception	Exception	Exception
53	Outgoing calls barred within CUG	Exception	Exception	Exception	Exception
55	Incoming calls barred within CUG	Exception	Exception	Exception	Exception
57	Bearer capability not authorized	Exception	Exception	Exception	Exception
58	Bearer capability not presently available	Exception	Exception	Exception	Exception
62	Inconsistency in designated outgoing access information and subscriber class	Exception	Exception	Exception	Exception
63	Service or option not available, unspecified	Exception	Exception	Exception	Exception
65	Bearer capability not implemented	Exception	Exception	Exception	Exception
66	Channel type not implemented (Q931 only)	Exception	Exception	Exception	Exception
69	Requested facility not implemented	Exception	Exception	Exception	Exception
70	Only restricted digital information bearer capability is available	Exception	Exception	Exception	Exception
79	Service or option not implemented, unspecified	Exception	Exception	Exception	Exception
81	Invalid call reference value (Q931 only)	Exception	Exception	Exception	Exception
82	Identified channel does not exist (Q931 only)	Exception	Exception	Exception	Exception

No.	Reason (see ITU-T Recommendation Q.850)	Terminating CallSetup	Terminating Stable Call		Terminating Call Clearing
			T-Alerting	T-Active	
83	A suspended call exists, but this call identity does not (Q931 only)	Exception	Exception	Exception	Exception
84	Call identity in use (Q931 only)	Exception	Exception	Exception	Exception
85	No call suspended (Q931 only)	Exception	Exception	Exception	Exception
86	Call having the requested call identity has been cleared (Q931 only)	Exception	Exception	Exception	Exception
87	User not member of CUG	Exception	Exception	Exception	Exception
88	Incompatible destination	Exception	Exception	Exception	Exception
90	Non-existent CUG	Exception	Exception	Exception	Exception
91	Invalid transit network selection	Exception	Exception	Exception	Exception
95	Invalid message, unspecified	Exception	Exception	Exception	Exception
96	Mandatory information element is missing	Exception	Exception	Exception	Exception
97	Message type non-existent or not implemented	Exception	Exception	Exception	Exception
98	Message not compatible with call state or message type non-existent or not implemented	Exception	Exception	Exception	Exception
99	Information element/parameter non-existent or not implemented	Exception	Exception	Exception	Exception
100	Invalid information element contents	Exception	Exception	Exception	Exception
102	Recovery on timer expire	Exception	Exception	Exception	Exception
103	Parameter non-existent or not implemented, passed on	Exception	Exception	Exception	Exception
110	Message with unrecognized parameter, discarded	Exception	Exception	Exception	Exception
111	Protocol error, unspecified	Exception	Exception	Exception	Exception
127	Interworking, unspecified	Exception	Exception	Exception	Exception
NOTE 1: The transition to the indicated DP is valid only for Present_Call. PIC.					
NOTE 2: The transition to the indicated DP is valid except for the Authorized_Termination_Attempt PIC.					
NOTE 3: The following rule applies for the stable call phase: The receipt of any cause value received during the stable call phase (O-Active/T_Active) shall in such states be mapped to O/T Disconnect DP.					
NOTE 4: The following rule applies for O/T-CallSetup, SendCall and O/T-Alerting: The receipt of any cause value received in forward direction during these call phases shall be mapped to O/T Abandon DP.					

6.4.6   BCSM Detection Point

Certain basic call and connection events may be visible to IN service logic instances.
DPs are the points in call processing at which these events are detected.
DPs for the BCSM are identified in the description of the "BCSM Model".

A DP can be armed in order to notify an IN service logic instance that the DP was encountered, and potentially to allow the IN service logic instance to influence subsequent call processing. If a DP is not armed, the CCF/SSF continues call processing without SCF involvement.

DPs are characterized by the following four attributes:

a) Arming/disarming mechanism:

The mechanism by which the DP is armed. A DP may be statically armed or dynamically armed. A DP is statically armed through SMF service feature provisioning. The ability of an SCF to statically arm or disarm a DP is allowed by the SCF using the ManageTriggerData operation. For a statically armed/disarmed DP the status remains unaffected until explicitly modified by the SMF or the SCF. A DP is dynamically armed by the SCF within the context of a call-associated IN service relationship.

The following DP disarming rules apply to dynamically armed DPs:

- If an armed EDP is met, then it is disarmed.

- If an EDP is met that causes the release of the related leg, then all EDPs related to that leg are disarmed.

- If a call is released, then all EDPs related to that call are disarmed.

b) Criteria:

In addition to the condition that a DP be armed, conditions that must be met in order to notify the SCF that the DP was encountered.

c) Relationship:

Given that an armed DP was encountered and DP criteria are met, the SSF may provide an information flow via a relationship:

- If this relationship is between the CCF/SSF and the SCF for the purpose of call/service logic processing, it is considered to be an IN service relationship.

With respect to an IN service relationship, the information provided by the SSF to the SCF on encountering a DP (TDP--R) may initiate a new relationship. This new relationship may be within the context of an existing relationship.

- If this relationship is between the CCF/SSF and the SCF or SMF for management purposes, it is considered to be a **service management relationship**. This relationship is outside the scope of this capability set.

d) Call processing suspension:

Given that an armed DP was encountered and DP criteria are met for an IN service relationship, the SSF may suspend call processing to allow the SCF to influence subsequent call processing. When call processing is suspended, the SSF request instructions from the SCF, and waits for a response. When call processing is not suspended, the SSF sends an information flow notifying the SCF that a DP was encountered, and does not expect a response. This attribute is set by the same mechanism that arms the DP.

Based on these attributes, four types of DPs are identified.

The DP types are:

1) Trigger detection point - Request (TDP-R);

2) Trigger detection point - Notification (TDP-N);

3) Event detection point - Request (EDP-R);

4) Event detection point - Notification (EDP-N).

The TDP-N is NOT supported.

These DP types are defined by the DP attribute values in table 8:

BCSM DPs may be any one of these DP types. DP processing for each DP type is described within clause 6 "DP Handling" and illustrated in figure 14 "Example: DP processing for each DP type".

Table 8: BCSM DP Types

DP type	Arming mechanism	Criteria	IN service relationship	Suspension	Service feature examples
TDP-R	Static	Specific to DP	Initiates relationship	Yes	All
EDP-R	Dynamic	None	Within context of existing relationship	Yes	Call distribution, call re-routing distribution
EDP-N	Dynamic	None	Within context of existing relationship	No	Charging for any service feature, call logging, call queuing

6.4.7   DP Criteria

As stated in previous clause "BCSM Detection Points" DP criteria are conditions that must be met in order to notify the SCF that the DP was encountered. These criteria can be assigned to a DP from the viewpoint of range of effectiveness, as identified by the trigger categories.

The following categories are DP trigger categories, as applicable for a given DP:

- **Individual -based** (also denoted "subscribed or line -based").

    This type of trigger category applies to each subscriber line or trunk line.
    For example, SCF processing is invoked when user A makes call origination. This trigger could be said to be specific for user A.

- **Group-based.**

    This type of trigger category applies to a certain trunk group of lines or users including Private Facility Group.
    For example, when a call origination from any user in a certain centrex group should invoke SCF processing the trigger should apply to that specific centrex group.

- **Switch-based** (also denoted "Office-based").

    This type of trigger category applies to the switch, that is the whole office. Any calls generated in the switching system will be subject to this trigger.
    For example, any call which makes access to the registered Freephone number is triggered and SCF processing is invoked.

The following criteria are **DP trigger criteria**, as applicable for a given DP:

- trigger assigned (unconditional/conditional on other criteria);

- class of service;

- specific B-channel identifier;

- specific digit strings;

- feature codes (e.g. *XX, #);

- prefixes (e.g. 0+, 00+, 0–, 00–, 011, 01, 1+);

- access codes (e.g. 8+) for customized numbering plan;

- specific abbreviated dialling strings for customized numbering plan;

- specific calling party number strings;

- specific called party number strings;

- nature of address (e.g. subscriber significant number, national significant number, international number);

- bearer capability;

- feature activation/indication (unconditional/conditional on specific feature patterns);

- facility information (unconditional/conditional on specific facility information patterns);

- cause (unconditional/conditional on specific cause patterns);

- USISeviceIndicator value (unconditional/conditional on specific value that identifies an IN service/service feature).

With respect to the DP criteria listed above, note that these DP criteria only apply to TDPs. DP criteria for Event Detection Points (EDPs) are addressed by the RequestReportBCSMEvent information flow.

In addition, note that one or more DP criteria may be applicable at a given DP.

The assignment of DP criteria to a TDP and the combinations of DP criteria applicable at a given DP continue to evolve. Further DP criteria and specific assignment of DP criteria to TDPs/EDPs may evolve through future capability sets.

NOTE 1:   Further that the assignment of DP criteria to a TDP on either a individual, group or switch basis may have an impact on the memory and real-time performance requirements of the CCF/SSF.

NOTE 2:   The applicability of DP criteria at a given DP depends on when call processing information is available and how long it is retained.

If network and service providers plan to implement IN CS-3 services in a multi-supplier environment, they should consider formulating such requirements to ensure consistent implementations across supplier equipment.
Such requirements should be considered carefully so as not to adversely impact memory and real-time performance aspects of CCF/SSF processing.

The DP criteria are defined below, as applicable to a given TDP:

1) Trigger assigned (see note 3) (unconditional/conditional on other criteria):

NOTE 3: It is possible that some DPs are always conditional.

- An indicator of the armed/disarmed status of a TDP assigned on an individual, group, or switch basis.

The trigger assigned criterion can be used by itself or in conjunction with other criteria at a TDP. If the trigger assigned criterion is unconditional at a TDP, then it is used by itself - no other DP criterion needs to be satisfied at the TDP before informing the SCF that the TDP was encountered. If the trigger assigned criterion is conditional at a TDP, then it is used in combination with other criteria at the TDP - all of the other DP criteria in the combination need to be satisfied before informing the SCF that the TDP was encountered.

Applies at all DPs (all DPs can be provisioned as TDPs).

2) Class of Service

- This is either a (i) customer class of service, (ii) trunk class of service, or (iii) private facility class of service;
(i) customer class of service is a code that identifies all attributes of a line that require distinctive call processing treatment (e.g. for party lines and coin lines),
(ii) trunk class of service is a code that identifies attributes of a trunk group such as type of signalling used; and
(iii) private facility class of service is a code that identifies attributes of a private trunk group such as type of signalling used and flash repeat capability.

Originating access (user/network) class of service is available at the Origination_Attempt DP and could be applicable at any of the originating DPs.

Terminating access (user/network) class of service is available at the Termination_Attempt DP and could be applicable at any of the terminating DPs.

3) Specific B-channel identifier

- An identifier of the specific B-channel on an ISDN interface from which a call attempt has originated or to which a call attempt is to be terminated.

A-party B-channel identifier is available at the Origination_Attempt DP for a party served by an ISDN interface only and could be applicable all originating DPs. B-party B-channel identifier is available during the Select_Facility PIC after an idle terminating facility has been selected for a party served by an ISDN interface only and could be applicable at the Facility_Selected_and_Available, T_No_Answer, T_Answer, T_Mid_Call, T_Suspend, T_Re_Answer and T_Disconnect DPs and at the T_Abandon DP (only after an idle terminating facility has been selected).

4) Specific digit strings

- A string of digits that must match collected digit strings for numbering plans in which a variable number of digits are to be collected. It could be zero or more digits (e.g. to trigger on "off-hook delay").

The string of digits should be consistent with the structure of the dialling plan and should be possible to administer. For example, the network provider may specify the first N digits where N is consistent with the structure of the E.164 numbering plan, or any other appropriate numbering plan.

Collected digit strings can be available at the Origination_Attempt DP for a party served by an ISDN interface using en bloc sending and at the Collected_Info DP for a party served by a non-ISDN line. Since collected digit strings are not analysed until the Analyse_Information PIC (except to determine if a sufficient number of digits have been collected), this criteria could be applicable at the Analysed_Info DP and beyond. The Analysed_Info DP [mandatory] and all those originating DPs that may be encountered after Analysed_Info [optional] are proposed since not all SSP suppliers may retain this information for the duration of the call/attempt.

- collected digit string can be available at the Origination_Attempt DP through ISUP signalling for an SS7 trunk.

- collected digit string can be available at the Collected_Info DP for a party served by a conventional trunk (e.g. non- SS7), ISDN interface using overlap sending, and private facilities.

5) *Feature codes* (e.g. *XX, #)

- A vertical service code, such as a "#" or a two-digit or three-digit code preceded by "*" or "11", that precedes any subsequent digit collection (e.g. according to the "normal numbering plan").

Feature codes can be available at the Origination_Attempt DP for a party served by an ISDN interface using en bloc sending or through ISUP signalling for an SS7 trunk, and can be available at the Collected_Info DP for non-ISDN lines and private facilities. Since collected digit strings are not analysed until the Analyse_Information PIC (except to determine if sufficient information has been collected), this criteria could be applicable at the Analysed_Info DP and beyond. The Analysed_Info DP [mandatory] and all those originating DPs that may be encountered after Analysed_Info [optional] are proposed since not all SSP suppliers may retain this information for the duration of the call/attempt.

Feature codes can be available at the Collected_Info DP for a party served by an ISDN interface using overlap sending.

6) *Prefixes* (e.g. 0+, 00+, 011, 01, 1+)

- A string of digits that are not feature codes or access codes and which precede any subsequent digit collection (e.g. according to the "normal numbering plan").

Prefixes can be available at the Origination_Attempt DP for a party served by an ISDN interface using en bloc sending, and can be available at the Collected_Info DP for non-ISDN lines, conventional trunks, and private facilities. Since collected prefix information is not analysed until the Analyse_Information PIC (except to determine if sufficient information has been collected), this criteria could be applicable at the Analysed_Info DP and beyond. The Analysed_Info DP [mandatory] and all those originating DPs that may be encountered after Analysed_Info [optional] are proposed since not all SSP suppliers may retain this information for the duration of the call/attempt.

Prefixes can be available at the Collected_Info DP for a party served by an ISDN interface using overlap sending.

7) *Access codes* (e.g. 8+) for customized numbering plan

- A string of digits in a customized numbering plan that matches access codes such as attendant access codes, access codes to escape to the public network, access codes to access a private facility, access codes to access a private network, and feature access codes.

Access codes can be available at the Origination_Attempt DP for a party served by an ISDN interface using en bloc sending, and can be available at the Collected_Info DP for non-ISDN lines and private facilities. Since collected access codes are not analysed until the Analyse_Information PIC (except to determine if sufficient information has been collected), this criteria could be applicable at the Analysed_Info DP and beyond. The Analysed_Info DP [mandatory] and all those originating DPs that may be encountered after Analysed_Info [optional] are proposed since not all SSP suppliers may retain this information for the duration of the call/attempt.

Access codes can be available at the Collected_Info DP for a party served by an ISDN interface using overlap sending.

8) Specific abbreviated dialling strings for customized numbering plan

- An abbreviated called party number in a customized numbering plan that must match collected address information.

Abbreviated address information can be available at the Origination_Attempt DP for a party served by an ISDN interface using en bloc sending, and at the Collected_Info DP for a party served by a non-ISDN line or private facilities. Since collected address information is not analysed until the Analyse_Information PIC (except to determine if sufficient information has been collected), this criteria could be applicable at the Analysed_Info DP and beyond. The Analysed_Info DP [mandatory] and all those originating DPs that may be encountered after Analysed_Info [optional] are proposed since not all SSP suppliers may retain this information for the duration of the call/attempt.

Specific abbreviated dialling strings can be available at the Collected_Info DP for a party served by an ISDN interface using overlap sending.

9) Specific calling party number strings

- A string of digits that must match the calling party number, which is a local, national, or international E.164 number or a number in a customized numbering plan. If a call has been forwarded, the calling party number is the number of the original calling party.

The calling party number is available at the Origination_Attempt DP in the originating BCSM and the Termination_Attempt DP in the terminating BCSM for a call originating from a non-ISDN line, ISDN interface, and can be available at the Origination_Attempt DP and the Termination_Attempt DP for SS7 trunks. This criterion could be applicable at all DPs.

10) Specific called party number strings

- A string of digits that must match the called party number, which is either a local, national, or international E.164 number, or a number in a customized numbering plan; the latter is not supported by SS7 or conventional trunks.
  If a call has been forwarded, the called party number is the number of the party that the call is forwarded to.
  The called party number can be available at the Origination_Attempt DP for a party served by an ISDN interface using en bloc sending or for an SS7 trunk, and can be available at the Collected_Info DP otherwise.
  Since collected address information is not analysed until the Analyse_Information PIC (except to determine if sufficient information has been collected), this criteria could be applicable at the Analysed_Info DP and beyond, and at all terminating DPs. In the originating BCSM, the Analysed_Info DP [mandatory] and all those originating DPs that may be encountered after Analysed_Info [optional] are proposed. No specific proposals are made for the DPs in the terminating BCSM.

11) *Nature of address* (e.g. Subscriber Significant Number, National Significant Number, International Number)

- An indicator of whether the called party number is a private, local (or subscriber), national, or international number.
  The nature of address is available at the Analysed_Info DP. This criteria could be applicable at the Analysed_Info DP and beyond. The Analysed_Info DP [mandatory] and all those originating DPs that may be encountered after Analysed_Info [optional] are proposed since not all SSP suppliers may retain this information for the duration of the call/attempt.

12) Bearer capability

- An indicator of the bearer capability as defined in the applied signalling system, e.g. ISUP/DSS.1
  The bearer capability information is available at the Origination_Attempt DP. This criteria could be applicable at all DPs.

13) *Feature activation/indication* (unconditional/conditional on specific feature patterns)

- In a local exchange only, a feature activation/indication on an ISDN interface or that is detected at the Mid_Call DP (e.g. "hook-flash", #, etc.) for ISDN and non-ISDN lines that can be sent in conjunction with or preceding other address/digit collection.
  A feature activation/indication can be available at DPs in the originating BCSM as indicated in table 9.

14) *Facility information* (unconditional/conditional on specific facility information patterns)

- A match on the Facility Information Element contained in a signalling message as defined in DSS1 and ISUP. Applicable DPs can be determined by mapping signalling messages to the BCSM and are outside the scope of this capability set.

15) *Cause* (unconditional/conditional on specific cause patterns)

- A match on the Cause value contained in a signalling message as defined in DSS1 and ISUP or an indicator of the cause of specific events of interest.
  See "Cause to DP Mapping Tables" to identify the cause values possible as DP criteria for IN CS-3 services from the complete list of cause values specified.
  This criteria is applicable at the identified DPs.

16) S*pecific value of the UNI/NNI* (DSS.1/ISUP) parameter mapping to a given USI ServiceIndicator value

- A value that identifies an IN service/service feature.

DP criteria assignment to a TDP is dependent on the information available at that DP.

The following two tables (tables 9 and 10) denote applicability of DP criteria to all the DPs.

The entries in the table can be of one of the following trigger categories:

- Individual-Based (Subscriber line or Trunk line);

- Group-Based (including Private Facility Group);

- Switch Based (also denoted Office Based).

Table 9: Originating DP Criteria

O_BCSM: DP Criteria	Originating DP															
	ARF	OA	OAD	OAA	CI	AI	RSF	OCPB	ONA	OTS	O Ans	OMC	OS	ORA	OD	OAb
Trigger Assigned	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Class of Service	O	X	O	O	O	O	O	O	O	O	O	O	O	O	O	O
Specific Calling Party Number (note 4)	O	X	O	O	O	O	O	O	O	O	O	O	O	O	O	O
Bearer Capability (note 5)	O	X	O	O	O	O	O	O	O	O	O	O	O	O	O	O
Specific B-channel Identifier	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O
Specific Digit String (notes 1 and 6)	O	O	O	O	X	X	O	O	O	O	O	O	O	O	O	O
Feature Code (note 1)	O	O	O	O	X	X	O	O	O	O	O	O	O	O	O	O
Prefixes (note 1)	O	O	O	O	X	X	O	O	O	O	O	O	O	O	O	O
Access Codes (note 1)	O	O	O	O	X	X	O	O	O	O	O	O	O	O	O	O
Called Party Number (note 1)	O	O	O	O	X	X	O	O	O	O	O	O	O	O	O	O
Specific abbreviated dialling string (note 1)	O	O	O	O	O	X	O	O	O	O	O	O	O	O	O	O
Nature of Address	O	-	-	-	-	X	O	O	O	O	O	O	O	O	O	O
Feature Activation (note 3)	X	-	-	-	-	X	X	X	X	X	X	X	X	X	X	X
Facility Information (note 2)	-	-	-	-	-	X	-	-	-	X	X	X	-	-	-	-
Cause	-	-	-	-	-	-	X	X	X	-	-	-	-	-	X	X
USISService-Indicator	O	X	X	O	O	O	O	O	O	O	O	O	O	O	O	O
X: Applicable	-: Not Applicable							O: Optional								

NOTE 1: Same type of trigger requiring analysis of a specific number of received digits. The analysis can be based on the complete number of received digits or can be based on a predefined number of digits starting from the most significant digit of the received information.

NOTE 2: A match on the Facility Information Element contained in a signalling message as defined in DSS1 and ISUP.

NOTE 3: In a local exchange only. The BCSM has to analyse (if facility is allowed, stored as Class of Service attribute) the received information and has to initiate an IN trigger if required. A feature activation/indication can be available at all DPs in the originating BCSM for a party served by an ISDN interface and can be available at the O_Mid_Call DP in the originating BCSM for a party served by a non-ISDN line. A feature activation/indication can be available at the T_No_Answer, T_Answer, T_Mid_Call, T_Suspend, T_Re_Answer, T_Disconnect and T_Abandon DPs in the terminating BCSM for a party served by an ISDN interface and can be available at the T_Mid_Call DP in the terminating BCSM for a party served by a non-ISDN line.

NOTE 4: The analysis should not be based on the complete calling party number, it shall be based on a predefined number of digits, starting from the most significant digit of the calling party number.

NOTE 5: Interpretation of Bearer Capability as optional for all DPs other than Origination_Attempt needs further clarification (e.g. Origination_Attempt DP mandatory means Termination_Attempt DP mandatory). Further, B-channel selection does not appear as a DP-criteria in the table because specific selection of B-channel by the user is outside the scope of this capability set. The network can override user selection of B-channel to be used.

NOTE 6: Digit strings, Feature codes, Prefixes, Access codes, Called party numbers & Abbreviated dialling strings can be available at the Origination_Attempt DP for a party served by an ISDN interface using en bloc sending or through ISUP signalling for an SS7 trunk and can be available at the Collected_Info DP for non-ISDN lines and private facilities.

The DPs in table 9 are abbreviated as follows:

OA =   Origination_Attempt
OAA =   Origination_Attempt_Authorized
OAD =   Origination_Attempt_Denied
CI =   Collected_Info
AI =   Analysed_Info
ARF =   Authorize_Route_Failure
RSF =   Route_Select_Failure
OCPB =   O_Called_Party_Busy
ONA =   O_No_Answer
OTS =   O_Term_Seized
OAns =   O_Answer
OMC =   O_Mid_Call
OS =   O_Suspend
ORA =   O_Re_Answer
OD =   O_Disconnect
OAb =   O_Abandon

Table 10: Terminating DP Criteria

T_BCSM: DP Criteria	Terminating DP												
	TA	TAD	TAA	TB	FSA	TNA	CA	TAns	TMC	TS	TRA	TD	Tab
Trigger Assigned	X	X	X	X	X	X	X	X	X	X	X	X	X
Class of Service	X	X	O	O	O	O	O	O	O	O	O	O	O
Specific Calling Party Number (note 4)	X	X	O	O	O	O	O	O	O	O	O	O	O
Bearer Capability (note 5)	O	O	O	O	O	O	O	O	O	O	O	O	O
Specific B-channel Identifier	-	-	-	-	O	O	O	O	O	O	O	O	O
Specific Digit String (note 1)	O	O	O	O	O	O	O	O	O	O	O	O	O
Feature Code (note 1)	-	-	-	-	-	-	-	-	-	-	-	-	-
Prefixes (note 1)	-	-	-	-	-	-	-	-	-	-	-	-	-
Access Codes (note 1)	-	-	-	-	-	-	-	-	-	-	-	-	-
Called Party Number (note 1)	O	O	O	O	O	O	O	O	O	O	O	O	O
Specific abbreviated dialling string (note 1)	-	-	-	-	-	-	-	-	-	-	-	-	-
Nature of Address	O	O	O	O	O	O	O	O	O	O	O	O	O
Feature Activation (note 3)	-	-	-	-	-	X	X	X	X	X	X	X	X
Facility Information (note 2)	-	-	-	-	-	-	X	X	X	-	-	-	-
Cause	-	-	-	X	-	X	-	-	-	-	-	X	X
USIService-Indicator	X	X	O	O	O	O	O	O	O	O	O	O	O
X: Applicable	-: Not Applicable							O: Optional					

NOTE 1: Same type of trigger requiring analysis of a specific number of received digits. The analysis can be based on the complete number of received digits or can be based on a predefined number of digits starting from the most significant digit of the received information.

NOTE 2: A match on the Facility Information Element contained in a signalling message as defined in DSS1 and ISUP.

NOTE 3: In a local exchange only. The BCSM has to analyse (if facility is allowed, stored as Class of Service attribute) the received information and has to initiate an IN trigger if required. A feature activation/indication can be available at all DPs in the originating BCSM for a party served by an ISDN interface and can be available at the O_Mid_Call DP in the originating BCSM for a party served by a non-ISDN line. A feature activation/indication can be available at the T_No_Answer, T_Answer, T_Mid_Call, T_Suspend, T_Re_Answer, T_Disconnect and T_Abandon DPs in the terminating BCSM for a party served by an ISDN interface and can be available at the T_Mid_Call DP in the terminating BCSM for a party served by a non-ISDN line.

NOTE 4: The analysis should not be based on the complete calling party number, it shall be based on a predefined number of digits, starting from the most significant digit of the calling party number.

NOTE 5: Interpretation of Bearer Capability as optional for all DPs other than Origination_Attempt needs further clarification (e.g. Origination_Attempt DP mandatory means Termination_Attempt DP mandatory). Further, B-channel selection does not appear as a DP-criteria in the table because specific selection of B-channel by the user is outside the scope of this capability set. The network can override user selection of B-channel to be used.

The DPs in table 10 are abbreviated as follows:

    TA =   Termination_Attempt
    TAA =   Termination_Attempt_Authorized
    TAD =   Termination_Attempt_Denied,
    TB =   T_Busy
    FSA =   Facility_Selected_and_Available
    TNA =   T_No_Answer
    CA =   Call_Accepted
    TAns =   T_Answer
    TMC =   T_Mid_Call
    TS =   T_Suspend
    TRA =   T_Re_Answer
                    TD = T_Disconnect
    Tab =   T_Abandon

If a criteria is marked with an "X" for a Detection Point, then this means that the criteria specific information associated with the trigger criteria shall be kept available until the DP is reached. If a criteria is marked with an "O" for a Detection Point, then this means that it is implementation dependent if the criteria specific information is still present at that DP because not all suppliers may retain this information for the duration of the call/attempt. If the information is still present, the treatment is the same as for a criteria marked with an "X".

6.5 Feature interactions manager (FIM)/call manager (CM)

A brief description of FIM/CM components in the CCF/SSF model regarding provided functionality and service logic program instance interaction aspects are provided below.

FIM functionality:

a) The FIM should provide a service logic program instance selection mechanism to determine which service logic program instance to invoke at a DP. This mechanism should select the appropriate IN service logic program instance or non-IN service logic program instance, and may block the invocation of any other service logic instances for that particular DP.

b) The FIM may not always allow simultaneously active IN and non-IN service logic instance for IN CS-3 that control the call/connection. There are both static and dynamic mechanisms of realizing this restriction. The static mechanism may involve service management functionality (e.g. via service provisioning), whereas the dynamic mechanism may involve more complex FIM capabilities. For IN CS-3, the simplest mechanism should be implemented.

c) The FIM should provide mechanisms to support simple, restricted service logic instance interactions between simultaneously active service logic instances from different SCFs acting on the same half call segment.

Call Manager (CM) functionality:

The connection control IN-SSM provides an SCF with an abstract view of a single two-party or multi-party call segment, or of a pair of associated call segments.

As such, the SCF can control multiple communication paths and connections, supported by multiple BCSMs.

Overall management of these various elements of call segments is provided by the Call Manager (CM) component in the CCF/SSF model.

The CM interacts with the BCM and IN-SM to:

a) coordinate event reporting among multiple BCSMs for a given IN connection control IN-SSM (e.g. event reporting when the same event is detected in multiple BCSMs simultaneously, such as "hook-flash", DTMF # or *XX, or when different events are detected in multiple BCSMs simultaneously, such as "hook-flash" from one party and "disconnect" from another);

b) coordinate the suspension and resumption of BCSM processing among multiple BCSMs for a given IN connection control IN-SSM (e.g. when an event is detected in a BCSM for which the BCM requires further instructions on how to proceed, processing of all BCSMs for that IN connection control IN-SSM may need to be halted);

c) enforce rules and restrictions applicable to an IN connection control IN-SSM (e.g. rules and restrictions on when and how the SCF can manipulate legs, associate a pair of call segments, and merge a pair of associated call segments).

Overall management of these various elements is provided by the Call Manager (CM) and Feature Interaction Manager (FIM) components in the CCF/SSF model and encompassed by the DP Handling subjects described below.

6.5.1 DP Handling

DP handling involves:

- traffic management actions (see call gapping and service filtering operation procedure descriptions);

- determining if DP criteria are met.(see this clause);

- handling service logic instance interactions when invoking new instances of IN and non-IN service logic (see description of FIM mechanism in this clause);

- and formulating messages to send to one or more SCFs (see this clause and Initial DP and event report information in operation procedure description clause (IN-SM).

The DP processing in the context of one single relationship and its relation to the components FIM and BCSM is illustrated in figure 14.
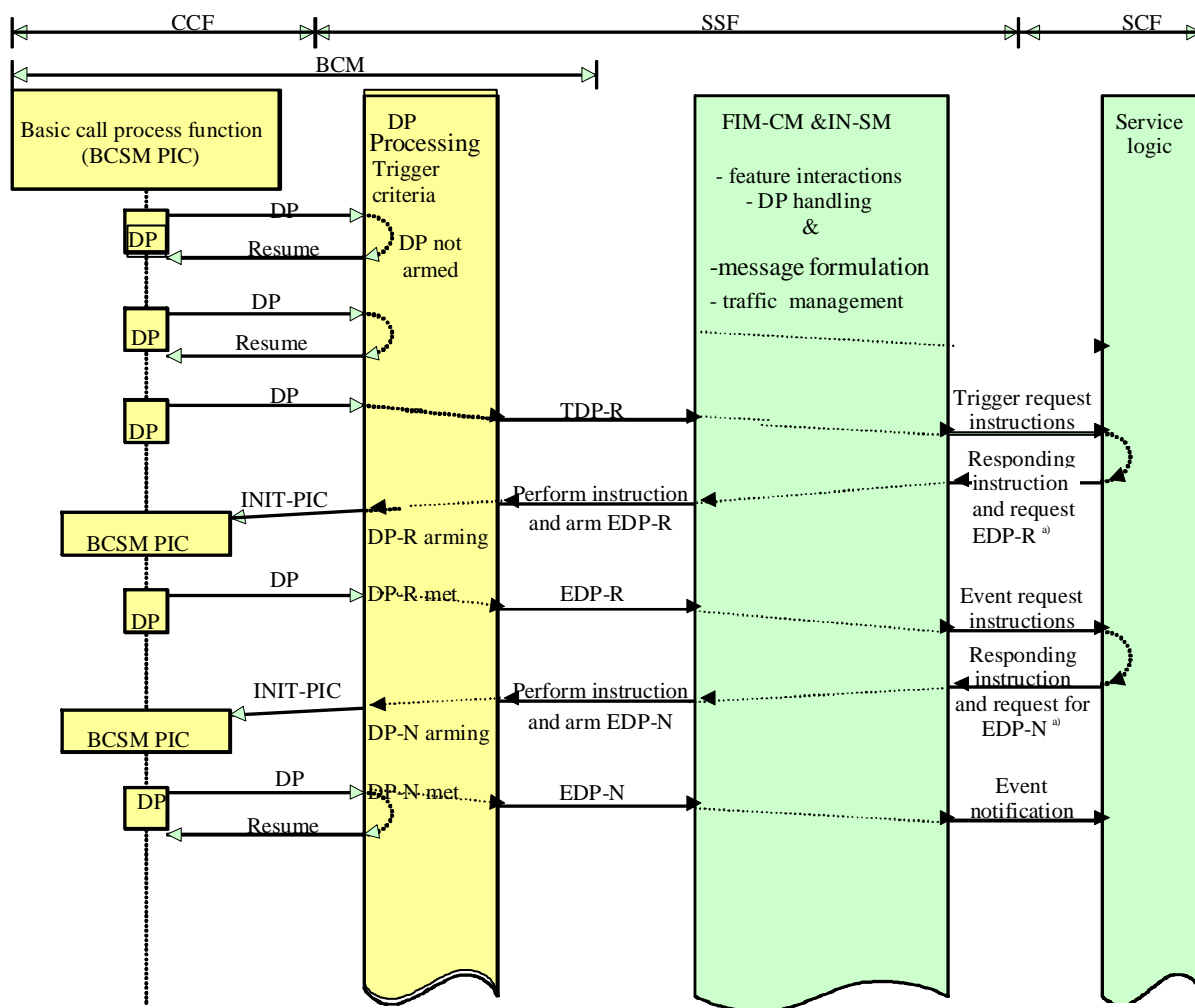
a) In this example, the responding instruction and request for EDP are shown togetherThese are independentinformation flows and may not be sent together in all cases.

DP   Detection point
TDP   Trigger detection point
EDP   Event detection point
R/N   Request/notification
PIC   Point in call

Figure 14: Example: DP processing for each DP type

6.5.1.1   FIM mechanisms

The FIM subjects include precedence and priority mechanisms to manage the invocation of instances of IN and non-IN service logic, and exclusion mechanisms to manage the invocation of new instances of IN service logic when existing instances of IN service logic are still active. These mechanisms described below cover the Single Point of Control (SPC) aspects:

a) *Precedence and priority*

   The assumptions that a DP may be armed as both a TDP and EDP, and that a DP may be armed with multiple criteria, each for the invocation of a different instance of IN service logic have previously been identified. The additional assumption that a DP may be armed for instances of non-IN service logic, in addition to instances of IN service logic, has also been identified. These assumptions, along with the constraints for IN CS-3 identified in the present document, form the basis of a set of precedence and priority rules that should be used when processing DP criteria. These rules are listed below:

   i)   when processing criteria for an armed DP, process criteria for a DP-Notification (DP-N) before a DP-Request (DP-R);

   ii) when processing criteria for a DP-N or a DP-R, process criteria for EDPs before TDPs;

   iii) when processing criteria for EDPs or TDPs, criteria processing rules for IN service logic and non-IN service logic must allow IN and non-IN service logic processing based upon priority of services;

   iv) when processing criteria for IN or non-IN service logic, process criteria in priority order, as provisioned through administrative procedures.

   Application of these rules may result in the following precedence ordering, with a priority ordering of multiple service logic instances at each level:

   - *EDP-N for an instance of IN service logic*
     A relationship exists with an SCF for an existing IN service logic program instance; the event detected at the DP is reported to the SCF in the context of the existing relationship and the next DP criteria is processed immediately. No response is expected from the SCF.

   - *EDP-N for an instance of non-IN service logic*
     The EDP is for an existing non-IN service logic program instance in the CCF/SSF; the event detected at the DP is reported to the non-IN FM and the next DP criteria is processed immediately. No response is expected from the non-IN FM.

   - *EDP-R for an instance of IN service logic*
     A relationship exists with an SCF for an existing IN service logic program instance; the event detected at the DP is reported to the SCF in the context of the existing relationship. Call processing is suspended and a response is expected from the SCF.

   - *EDP-R for an instance of non-IN service logic*
     The EDP is for an existing non-IN service logic program instance in the CCF/SSF; the event detected at the DP is reported to the non-IN FM. Call processing is suspended and a response is expected from the non-IN FM.

   - *TDP-R for an instance of IN service logic*
     The event detected at the DP is reported to the SCF via a new relationship. Call processing is suspended and a response is expected from the SCF.

   - *TDP-R for an instance of non-IN service logic*
     The TDP is for a non-IN service logic program instance in the CCF/SSF; the event detected at the DP is reported to the non-IN FM. Call processing is suspended and a response is expected from the non-IN FM.

   For those cases in which a response is expected and the response indicates that call processing should continue from the point at which it was suspended (i.e. the DP at which criteria were met and the event was reported), then the remaining DP criteria should be processed. If the response indicates that call processing should continue at a new point in call, then any remaining DP criteria at the point of suspension are not processed.

b) Exclusion

There is a mechanism in IN for managing exclusion of new instances of IN service logic program when existing instances of IN service logic program are still active. However, with this IN Capability Set Multiple Points of Control is supported. Rules are defined that allow more than one IN service logic instance at a time to control the same call segment (i.e. to send responses to the CCF/SSF). These rules supports multiple IN service logic instances simultaneously controlling the same call segment. In addition, item a) above describes precedence and priority rules for processing DP criteria for service logic instances. These rules identify that service logic instances at a lower precedence level or priority may not be invoked, depending on the disposition of previous service logic instances. This implies that DP criteria for multiple service logic instances at the same DP can be ordered in such a way as to manage this exclusion.

6.5.1.1.1   Service logic instance interactions considerations

It is recognized that services provided by an IN-structured network will be composed of one or more service features, which are constructed from one or more reusable units of capabilities (e.g. SIBs) provided to users by the network. It is also recognized that one or more service features may be simultaneously active on a single call. Finally, it is recognized that both IN service features and non-IN service features may be simultaneously active on a single call. A service feature interactions mechanism is needed to manage the potential interactions (both desirable and undesirable) between such service features. Given that these service features are realized by service logic instances, this mechanism needs to be described in terms of rules and procedures relative to triggering, compatibility, precedence, invocation, execution, and event reporting for multiple service logic instances. This clause addresses the static and dynamic aspects of service logic instance interactions management, as well as mechanisms for determining compatibility and precedence.

a) Static and dynamic aspects

There are two aspects to service logic instance interactions management, to include the static and dynamic aspects. These two aspects are discussed below.

- Static aspects

   The static aspects of service logic instance interactions management concerns the provision of service features to end users. To illustrate this, consider the following example: end user A already has service feature X, and it is known that service feature X and service feature Y are mutually incompatible; if an attempt is made to provision the invocation of a service logic instance for service feature Y to end user A using OA&M procedures, then this attempt should be rejected.

- Dynamic aspects

   There are three items to be considered under the dynamic aspects of service logic instance interaction management.

   ▪ If at a particular DP there is more than one service logic instance which can be invoked, then a decision must be made as to which of these service logic instances will be invoked first (i.e. service logic instance selection).

   ▪ If a service logic instance can be invoked, then a decision must be made as to whether or not the new service logic instance is compatible with any service logic instances already active on the same call segment.

   ▪ If the new service logic instance is compatible with any service logic instances already active on the same call segment, then a decision must be made as to its precedence for call processing events (such as signalling messages) with respect to other active service logic instances; if the new service logic instance is incompatible, it should be blocked.

   For the latter two items, there are at least two potential approaches to service logic instance interactions management.

   ▪ The first approach is to make decisions as part of DP processing; with this approach, decisions about service logic instance compatibility and precedence are made *before* a service logic instance is invoked.

   ▪ The second approach is to make decisions independent of DP processing; with this approach, decisions about service logic instance compatibility and precedence are made *after* a service logic instance is triggered.

The first approach is simpler, though restrictive, since it can prevent service logic instances from being invoked, only requiring the management of a limited number of service logic instance interactions. The second approach is more complex, though flexible, since it does not prevent service logic instance from being invoked, thus requiring a mechanism that can manage all possible service logic instance interactions. Due to this complexity, the second approach is considered beyond the scope of IN CS-3.

b) *Mechanisms for determining compatibility and precedence*

At present, knowledge concerning the compatibility of service features and their precedence is "hard-coded" in to the CCF/SSF. This mechanism relies on specifying each possible interaction for every possible combination of service features. As the number of service features gets large (one of the aims of IN), this specification quickly becomes complex, complicating the task of the service designer. Furthermore, as each new service feature is added, its many possible interactions must be identified and specific rules and data must be introduced into the CCF/SSF or SCF to specify how each interaction is to be resolved.

A more general mechanism than "hard-coding" would be a "data-driven" mechanism in which the service designer could specify service feature compatibility and precedence during service creation and provisioning. The service creation environment could provide the service designer with information about the specific service features for a particular subscriber, enabling the service designer to specify such things as which service features are blocked by a new service feature, the relative precedence of the new service feature to other service features, and the DP at which the service logic instance for the service feature should be invoked. The output of such a mechanism could be introduced directly into the CCF/SSF or SCF from the service creation environment.

The ultimate mechanism would be to use an expert system approach to reduce the burden on the service designer.

The existing mechanisms for service logic instance interactions management will have to be used beyond what is described herein for IN-IN service logic instance interactions in the CCF/SSF, and below item c), d), and for IN-non IN service logic instance interactions in the CCF/SSF. That is, the interactions between service logic instances (both IN and non-IN) will have to be specified as part of the service feature description, with vendor-specific mechanisms to resolve remaining interactions in the specified manner. In addition, it may be possible to adopt a data-driven approach if mechanisms can be incorporated into the service creation environment to prompt the service designer for compatibility and precedence information, then download the appropriate data into the CCF/SSF or SCF. The expert system approach is considered to be beyond the scope of this IN Capability Set.

c) *IN and non-IN service logic instance interactions*

There are desirable and undesirable IN and non-IN service logic instance interactions in the CCF/SSF. Table 11 identifies these.

Table 11: IN and non-IN service logic instance interactions

		Non-IN		
		Connection Control (CC)	**Non-connection control**	
			Passing or using information	**Notification**
IN	**CC**		Must be independent	Cannot be independent	OK
	Non-CC **Request**	**Restricted** (e.g. translation)	• OK for passing information (e.g. CLID) or • Need precedence if using same info or same DP	OK
	Notification	OK	OK	OK

This table classifies IN and non-IN service logic instances first by whether or not they involve connection control (e.g. leg manipulation). Non-IN service logic instances that do not involve connection control are further classified by their involvement in call/service processing. This includes involvement in passing end-to-end information on a call (e.g. user-to-user information, calling number delivery) or using call-related information (e.g. for number translation), and involvement only in terms of receiving notification of call-related events (e.g. answer, disconnect). IN service logic instances are also further classified by their involvement in call/service processing. This includes involvement in terms of receiving requests and providing non-connection control instructions (e.g. Proceed call processing with new information), and involvement only in terms of receiving notification of call-related events. Based on these classifications, a matrix of interaction restrictions for IN can be developed, as reflected in table 11.

From the table, it is evident that IN service logic instances that involve connection control must be completely independent of non-IN service logic instances that involve connection control. This is a consequence of the single point of control constraint.
Further, it is evident that IN and non-IN service logic instances that only involve notification of events can interact with any other type of IN and non-IN service logic instances, since these do not involve any type of control. The remaining interactions are restricted as follows:

- *IN CC vs. non-IN passing or using information*
  In this case, the service logic instances cannot be processed independent of each other since IN service logic instances that involve connection control may prevent passing of end-to-end information by changing or interrupting connections.

- *IN non-CC request vs. non-IN CC*
  In this case, IN service logic instances are restricted to those that only manipulate basic call-related information (e.g. for destination number translation), and do not change the flow of basic call processing (e.g. given that basic call processing is suspended while waiting for IN call handling instructions, processing resumes from the point at which it was suspended when instructions are received). In this case, IN service logic instances can be invoked to enhance non-IN connection control [see discussion item d) below].

- *IN non-CC request vs. non-IN passing or using information*
  In this case, passing end-to-end information should be transparent to IN service logic instances. However, IN and non-IN service logic instances may be competing for the same call-related events or information. Straightforward precedence and exclusion mechanisms can be used to resolve this contention for IN. These mechanisms are described.

These restrictions are identified as guidelines to assist implementers in managing these types of interactions in a proprietary manner in those cases where mechanisms are not described in the present document.

d) *Applying "Type A" IN technology to "Type B" services*

There are some circumstances in which it will be possible to apply "Type A" IN technology to certain aspects of "Type" services. This applies to switch-based services in general, whether these services of "Type A" or "Type B", and to "Type B" services in general, whether these be switch-based or CS-N based.

"Type A" services are characterized as "single-ended" and "single point of control". It also happens that this IN Capability Set is limited to "single medium" (as opposed to "multi-media") services. By implication, "Type B" services differ from "Type A" services in at least one of the dimensions: (ends, points of control, media). Of main interest in the shorter term is variation of the number of ends affected. Some examples of "Type A" services are: Freephone, virtual private network (VPN), universal personal telecommunications (UPT), originating and terminating call screening, selective call forward on busy/do not answer, credit card calling, televoting, malicious call identification, and completion of call to busy subscriber. Currently defined "Type B" services are generally available using switch-based technology. It may be expected that equipment vendors will provide support and interworking of "Type A" and "Type B" services in their product portfolios. Such interworking will not necessarily be part of the standards for this IN Capability Set.

i) Situations when "Type A" capabilities may be used with "Type B" or switch-based services

- In circumstances where a request for a "Type B" or switch-based service requires a check to see whether such a service may be performed, "Type A" technology may be applied before proceeding with the service.

- In circumstances where several variations in a "Type B" or switch-based service are possible, a check to see which variation is to be performed may make use of "Type A" technology.

ii) Determining when to use "Type A" capabilities

- In the active phase of a call, certain means for gaining the attention of the exchange (e.g. switch-hook flash) are context specific. In these circumstances, the context needs to be considered first to determine whether a "Type A" service request should occur. For example, after receiving call waiting tone, a series of switch-hook flashes may be used to toggle between the two calls. In the absence of call waiting, a switch-hook flash may indicate a desire to add in a third party, with a subsequent switch-hook flash joining the three subscribers.

- From these two cases, it can be seen that some care needs to be taken in determining whether or not it is appropriate to launch a "Type A" service query. In the example described, it would not be appropriate once the call waiting tone has been applied, nor after the waiting call has been answered. In the second case, it would be appropriate to see what should be done. Some options that could be indicated to the switch might be: ignore the switch-hook flash, proceed with normal three party call, add-in fixed third party (e.g. supervisor), etc.

- Taking full advantage of this approach will require some extension of the SSF - SCF interface to include identification of the specific service and instruction to proceed or not with (standardized) services. The extent to which this can be standardized will depend on the time and resources available to do it as the standardization for the base capabilities to support "Type A" services proceeds.

iii) Examples of services augmented by "Type A" capabilities

- Conference dial-in authorization.

  In this service, only authorized parties may dial-in to a conference bridge. Conferencing is, in general, a "Type B" service in that more than one end is involved when another subscriber joins the conference.

  An SSF supporting a conference capability, on receiving a request to join the conference, may use "Type A" technology to query an SCF for a list of authorized participants. This list would be updated through an OA&M process as conference reservations are made, and would include such things as: conference timings, participant identification, charging to be applied, etc. This list could even be updated in real time as the conference proceeds so that previously excluded subscribers may join in as directed by the conference "owner" or Chairman.

  In this way, a substantial degree of security may be added to a conference, especially one that is regularly held and at which sensitive information might be discussed.

- Selective or distinctive call waiting.

  In order to determine whether a call waiting tone should be applied, the terminating exchange may consult an SCF for a screening list (inclusive or exclusive) to determine whether call waiting should be applied or whether alternative treatment should be given to the incoming call. In this way, "Type A" technology can be used to augment this service.

  In order to indicate certain special callers, a distinctive call waiting tone may be applied. "Type A" technology may be used to identify when this applies and, when there are several distinctive tones available, which should be applied. In this way, "Type A" technology can be used to augment this service.

6.5.1.2   DP and Event Distribution and Filtering

One general objective of IN CS-3 is to support Multiple Points of Control (MPC).
In this clause some MPC rules are defined, however, these rules does not cover all aspects of MPC.
Enhancements to these rules and TDP and EDP processing scenarios have to be provided with the next IN Capability Set.
For a given half-call either Single Point of Control (SPC) or MPC applies.
The decision whether SPC or MPC rules are to be used is done during triggering of the first service logic depending on trigger related data.

6.5.1.2.1   General Rules

Since a DP may be armed as a TDP and/or an EDP for the same call segment association (CSA), the BCM should apply the following set of rules during DP criteria processing:

Rule 1: At any DP, a specific trigger condition can only trigger one service logic program instance (SLPI) at a time.

Rule 2: If a DP is both armed as EDP and TDP, then the EDP processing has higher priority than the TDP processing since the EDP has been armed in an already existing SSF-SCF relationship.

> NOTE 1: This default rule may imply some assumptions on the SLP behaviour (e.g. the first SLP which receives a failure event should respond with a Continue or CWA operation in order to allow this event to be reported to a subsequent SLP).

The rules are listed in descending priority order.

A relationship remains as long as there is at least 1 armed EDP or a pending report (CallInformationReport, ApplyChargingReport), within a Call Segment for a Call Segment Association. A relationship terminates if there are no more EDPs armed and no pending reports (CallInformationReport, ApplyChargingReport) or the call clears. During a relationship, EDPs may be dynamically disarmed by the SCF, or are disarmed by the SSF as they are encountered, or when the call clears.

In addition:

- encountering of a TDP-R (i.e. sending of InitialDP from the SSF to the SCF) creates a new relationship;

- encountering of the last EDP-R within a CSA maintains the existing relationship even though no additional EDPs are armed.

When call processing is resumed the relationship will:

- continue as a relationship if there is at least 1 armed EDP or a pending report (CallInformationReport, ApplyChargingReport), within a Call Segment for a Call Segment Association;

- terminate the relationship if there are no EDPs armed or pending reports (CallInformationReport, ApplyChargingReport).

> NOTE 2: It is possible to cause infinite re-triggering during the processing of a call. Such an example is as follows:
>
>   1) User dials 555-1111.
>
>   2) Analysed_Information DP is armed as TDP-R, and the criteria is 555-1111.
>
>   3) SCF returns a Destination Routing Address of 555-1111 in the Connect operation.
>
>   4) The call will resume at the Analyse_Information PIC and then trigger at the Analysed_Information DP again. This effectively produces an infinite loop.

It should be noted that this loop can occur any time the response from a DP returns the call to a previous point in the BCSM. ITU-T Recommendation Q.1229 provides details as to how re-triggering can be avoided.

6.5.1.2.2   Single Point of Control (SPC) Rules

Rule 1: At any DP, processing of notifications - EDP-N - has higher priority than processing of requests - EDP-R and TDP-R. If several notifications exist, EDP-R and TDP-R are processed when all notifications have been processed.

Rule 2: If a DP is both armed as EDP and TDP, then the EDP processing has higher priority than the TDP processing since the EDP has been armed in an already existing SSF-SCF relationship.

Rule 3: If a DP is both armed as EDP-R and TDP-R, the EDP-R is first processed and, if the relationship is terminated as a result of the EDP-R processing, processing of the TDP-R is allowed.

The rules are listed in descending priority order. They are illustrated with the SDL diagram in figure 15.

Single point of control ensures that only one controlling SCF (service logic) exists. Single point of control is only guaranteed within a Call Segment Association.

6.5.1.2.3   Multiple Points of Control (MPC) Rules

DP Processing Rules:

For MPC the following DP arming rules and DP reporting rules applies for the FIM.

General Objective:

If there are more than one controlling SCF acting on the same Originating half call or Terminating half call, then the event detection point processing requested by any of the involved service logics (SLPIs) shall be performed in the same way as if triggering had occurred in different Originating BCSMs respectively in different Terminating BCSMs, which are separated by a Network Node interface.

As a network operators option the MPC rules specified may be replaced or enhanced by network operator specific rules based on service related data, e.g.:

- in case a DP is armed as EDP and TDP the usage of a precedence order to allow a TDP processing before and/or after EDP processing;

- usage of precedence order for EDP reporting.

For supporting both SPC and MPC, the FIM shall perform DP processing according to the following detailed processing rules:

1   TDP Reporting Rules

   1.1 At any DP, if processing of a TDP-R is allowed while there exists a SSF - SCF relationship, the TDP-R is reported only for the case the service compatibility check in the SSF is successful. The compatibility information from all previous invoked services has to be considered (received from SCF in the Connect, ContinueWithArgument, InitiateCallAttemt; or via signalling).

2   EDP-Arming Rules

   2.1 Each service logic program instance shall arm/disarm its own EDP-Rs and EDP-Ns independently from the other service logic program instances (SLPIs).

   2.2 If an EDP-R is armed for at least one SSF instance then the FIM will arm the EDP as EDP-R in the CCF independently whether a new SSF instance arms or disarms this EDP.

   2.3 If a EDP-N is armed by a new SSF instance and the EDP is not armed as EDP-R by another SSF then the EDP is armed by the FIM as EDP-N in the CCF.

   2.4 If an EDP-N is armed by a new SSF instance and the EDP is already armed as EDP-N by another SSF then the EDP is not armed a second time by the FIM as EDP-N in the CCF.

3   EDP-Reporting/Execution Rules

   3.1 EDP's reported from the CCF, caused by a forward event, shall be distributed by the FIM to the SSF instances in the forward triggering order of the SSF instances.

   3.2 EDP's reported from the CCF caused by a backward event will be reported by the FIM in the reverse triggering order of the SSF instances.

   3.3 In case of an EDP-R, reporting to a subsequent service logic program instance is only performed after the call processing has been resumed by a Continue or ContinueWithArgument operation.
   E.g. a received release message (O/T_Disconnect DP) would not be further propagated to the network or to the subsequent SSFs if the SCF answers the EDP-R with a new Connect operation.

4   Distribution of release events (ReleaseCall and DisconnectLeg) received from SCF.

   4.1 In a multiple points of control context, several service logic program instances (SLPIs) can release the call or a leg (via ReleaseCall or DisconnectLeg).

The release events received from a SCF shall be handled identical as if a basic network release message had been received (TDP/EDP processing).

5   Charging Rules

5.1 Charging event-Arming Rules:
Each service logic program instance shall arm/disarm its own charging events independently from the other service logic program instances.

5.2 Charging event Reporting/Execution Rules:

a) A charging event reported from the CCF caused by a forward event will be distributed by the FIM to the SSF instances in the forward triggering order of the SSF instances.

b) Charging events reported from CCF caused by a backward event will be reported by the FIM in the reverse triggering order of the SSF instances.

c) In case of a charging event armed in mode "interrupted", reporting to any subsequent service logic program instance is not performed.

5.3 Distribution of charging events received from SCF:

In a multiple points of control context, several service logic program instances can send charging instructions (in operation SendChargingInformation). The direction of the event is indicated by legID. The charging events received from a SCF will be handled identical as if a basic network signalling charging message had been received.

6   CPH Handling Rule:

6.1 When receiving an operation which leads to the creation of a new CS instance in a CSA where one CS instance already exists (e.g. SplitLeg, InitiateCallAttempt, MoveCallSegment (for the target CSA)), the operation has to be refused if for the same half call another CSA already exists with more than one CS inside or in case of single CS, the CS contains more than two connected joined legs (e.g. multi-party call segment).

7   USI information interaction

7.1 Distribution of usi information (usi operations) received from SCF.
In a multiple points of control context, several service logic program instances can send usi information to a user (via SendSTUI).
The usi information received from a SCF shall be handled identical as usi information received from the basic network.

7.2 Reception of usi information for the SCF.(ReportUTSI)

a) User information received from the CCF in the forward direction, shall be distributed by the FIM to the SSF instances in the forward triggering order of the SSF instances. If one service logic has armed this event for the given usi service indicator, it shall not be propagated to the subsequent SLPI's in the forward direction.

b) User information received from the CCF in the backward direction, shall be distributed by the FIM to the SSF instances in the reverse triggering order of the SSF instances. If one service logic has armed this event for the given usi service indicator, it shall not be propagated to the subsequent SLPI's in the backward direction.

6.5.1.3 TDP and EDP Processing Scenarios for Single Point of Control
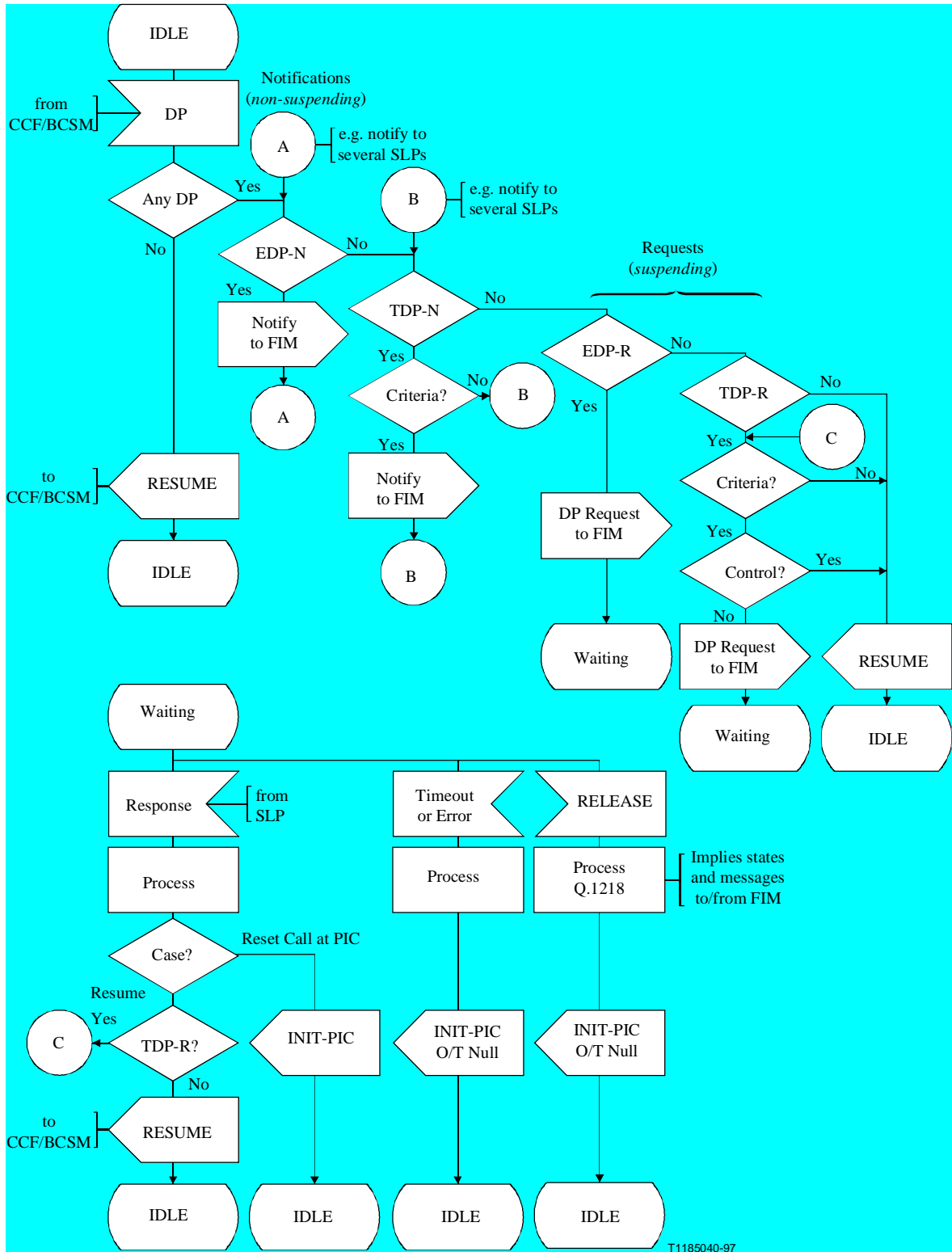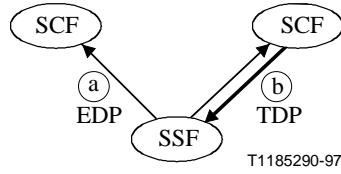
NOTE: TDP-N is not supported, i.e. TDP-N = NO branch shall always be selected.

Figure 15: Detection point processing for Single Point of Control

As a consequence of these rules, the BCM should support a number of TDP/EDP processing combinations to ensure Single Point of Control. These combinations are identified in table 12, along with three error combinations that should not occur.

These combinations applies only within one call segment association.

Table 12: Detection point processing combinations for Single Point of Control

Scenario	TDP type	EDP type	Existing relationship	Processing
1	Not armed	Not armed	Do not care	Continue
2	TDP-R	Not armed	No	Initiating DP request
3.a	TDP-R	Not armed	Yes	Continue (ignore TDP)
3.b	TDP-R	Not armed	Yes	Initiating DP request
5.a	Not armed	EDP-R	Yes	Subsequent DP request, if armed EDPs remaining, or terminating DP request, if last armed EDP
5.b	Not armed	EDP-R	Yes	Error case – Continue (ignore EDP)
6	Not armed	EDP-N	Yes	Subsequent DP notification, if armed EDPs remaining, or terminating DP notification, if last armed EDP
7	Not armed	EDP-R/N	No	Error case – Continue (ignore EDP)
11.a	TDP-R	EDP-N	Yes	Process a and b: a) For EDP, subsequent DP notification, process as scenario 6 b) Ignore TDP
11.b	TDP-R	EDP-N	Yes	Process a and b: a) For EDP, process as scenario 6 b) For TDP, initiating DP request, process as scenario 3.b
12	TDP-R	EDP-R/N	No	Error case – Ignore EDP and process TDP as scenario 2
13.a	TDP-R	EDP-R	Yes	Process EDP as scenario 5.a. If this EDP was the last of the previously established relationship, process the TDP afterwards. Otherwise the TDP is ignored
13.b	TDP-R	EDP-R	Yes	Error case – Ignore EDP and process TDP as scenario 3.b
NOTE:	Scenario 8-10 is not supported (refer to ITU-T Recommendation Q.1238).			

6.5.1.4 Implicit EDP Disarming Rules

Implicit EDP disarming rules are specified in the tables below for Originating BCSM and respectively Terminating BCSM. Each table specifies which EDPs should be disarmed (i.e. MonitorMode set to Transparent) if/when each EDP is encountered, irrespective of the EDP's MonitorMode (Transparent, NotifyAndContinue, or Request).

When EDP's armed with MonitorMode 'Request' (EDP-R's) are encountered, any implicit EDP disarming should take place before reporting the EDP and transiting the "FSM for CS" to the WFI state (if not already suspended in a WFI state).

If more than one BCSM instance is present in a single Call Segment and at least one of the BCSMs has encountered O_Answer DP/T_Answer DP then an originator release must be detected as a O_Disconnect DP/T_Disconnect DP event.

NOTE: The rules are designed for use in a Single Point of Control configuration and may require further enhancements if they were to be used in a Multiple Points of Control configuration. Enhancements to these rules in order to cover all aspects of MPC will have to be catered for in the next IN- Capability Set.

Table 13: O_BCSM: Implicit EDP Disarming Table

EDP Disarmed / EDP Encountered	1	2	3	4	5	6	7	8 C Leg	8 P Leg	9 C Leg	9 P Leg	10	19	20	24	28	29
1	#																X
2		#															
3			#														
4				#	#	#	#		#		#		#	#	#	X	
5				#	#	#	#		#		#		#	#	#	X	
6				#	#	#	#		#		#		#	#	#	X	
7				X	X	X	#					X (note 2)	X		X		
8 Controlling Leg								# (note 1)									
8 Passive Leg									# (note 1)								
9 Controlling Leg	#	#	#						#	#		#					
9 Passive Leg				#	#	#	#		#		#		#	#	#		
10	#	#	#						#	#		#					
19													#				
20														#			
24															#		
28				X	X	X	X		X		X		X	X	X	X	
29																	X

Table 14: T_BCSM: Implicit EDP Disarming Table

EDP Disarmed / EDP Encountered	12	13	14	15	16 C Leg	16 P Leg	17 C Leg	17 P Leg	18	21	25	26	27	30
12	#													X
13	#	#	#	#	#		#			#	#	#	#	
14	#	#	#	#	#		#			#	#	#	#	
15		X	X	#					X (note 2)				X	
16 Controlling Leg					# (note 1)									
16 Passive Leg		X	X	X		# (note 1)			X				X	
17 Controlling Leg		#	#	#	#		#			#	#	#	#	
17 Passive Leg	#					#	#	#						
18						#	#	#						
21										#				
25											#			
26												#		
27													#	
30														X

Legend:
Represents IN CS-1 (1995) compliant SSF EDP disarming (e.g. leg is released, EDP is encountered).
X Represents IN CS-3 SSF Implicit Disarming of EDP.
NOTE 1: Only the detected service code or the range to which the service code belongs is disarmed.
NOTE 2: O/T-Abandon DP is disarmed if it is the last armed DP and no other reports are pending; this allows the relationship to be closed.

Key to EDPs:

EDP1 - Origination_Attempt_Authorized
EDP2 - Collected_Information
EDP3 - Analysed_Information
EDP4 - Route_Select_Failure
EDP5 - O_Called_Party_Busy
EDP6 - O_No_Answer
EDP7 - O_Answer
EDP8 - O_Mid_Call
EDP9 - O_Disconnect
EDP10 - O_Abandon
EDP19 - O_Term_Seized
EDP20 - O_Suspend
EDP24 - O_Re-answer
EDP12 - Termination_Attempt_Authorized
EDP13 - T_Busy
EDP14 - T_No_Answer
EDP15 - T_Answer
EDP16 - T_Mid_Call
EDP17 - T_Disconnect
EDP18 - T_Abandon
EDP21 - T_Suspended
EDP25 - T_Re-answer
EDP26 - Facility_Selected_And_Available
EDP27 - Call_Accepted
EDP28 – Authorize_Route_Failure
EDP29 – Origination_Attempt_Denied
EDP30 – Termination_Attempt_Denied

6.6 IN-switching manager (IN-SM)

A brief description of the IN-SM is provided in EN 301 931-1.

The IN-SM centres an the IN-switching state model (IN-SSM) which provides a description of CCF/SSF IN call/connection processing in terms of IN call/connection states. Object-oriented techniques are used to describe the IN-SSM, based on the concepts and principles outlined in annex B of ITU-T Recommendation Q.1204.

The IN-SM supports IN call party handling capabilities.
The IN-SM subjects described in the following s include the IN-SM call party handling capabilities, IN-SSM, IN-SSM events that can be reported to active IN service logic instances, and SSF resource control.
A high-level description of these subjects is provided.

6.6.1 IN-switching state model (IN-SSM)

The IN-SSM provides an object-oriented finite state machine description of CCF/SSF IN call/connection processing in terms of IN call/connection states.
It provides a framework for describing the scope of view and control of CCF/SSF activities offered to an SCF. The extent to which the IN-SSM is visible to the SCF is defined by the INAP operations identified between the CCF/SSF and SCF. See clauses 11 and 12 providing the operation procedure description and the associated operation parameter descriptions.

IN call/connection states can be described in terms of the IN-SSM, which defines the set of CCF/SSF objects visible to the SCF.
Each IN-SSM instance provides the SCF with a limited aperture of visibility and influence into CCF/SSF IN call/connection processing. The objects that constitute the IN-SSM define this aperture of visibility and influence. These objects are abstractions of CCF/SSF resources accessible to the SCF.

There can be various types of IN-SSMs, each type defined by the objects that constitute it. For example, a Call Segment Association IN-SSM would contain objects that are abstractions of switching and transmission resources.
This clause focuses on such a Call Segment Association IN-SSM, though it is recognized that other types of IN-SSMs may exist for accessing other types of resources.

There can also be various subtypes of a particular IN-SSM type; each defined by a subset of, or restriction on the use of, the total set of objects in the IN-SSM type.
It is anticipated that IN-SSM subtypes will be identified to align with specific IN capability sets as they are defined.

A Call Segment Association (CSA) instance is created by the Service Switching Function Management Entity – Control (SSME-Control) as further described in clause 8, when an IN service logic instance is invoked that requires IN connection control. It is either created as a result of encountering a TDP in a BCSM that satisfies DP criteria, or is initiated by the SCF independent of encountering TDPs. A CSA instance is destroyed when the SCF informs the SSF that the IN service logic instance is completed or the CSA should be destroyed. The SSF can also initiate CSA destruction (e.g. during error or abnormal conditions).

The characteristics of CCF/SSF call processing represented by CSA objects are described below. These characteristics imply the attributes and functions related to CSA objects, to be reflected in the call processing messages/parameters defined for IN CS-3.

a) The IN Call Segment Association provides the SCF with an abstract view of an isolated portion of a call managed by a functionally separate portion of the CCF/SSF. This isolated portion of a call is referred to as a "half-call" or *call segment* (see CCF/SSF model - figure 1).
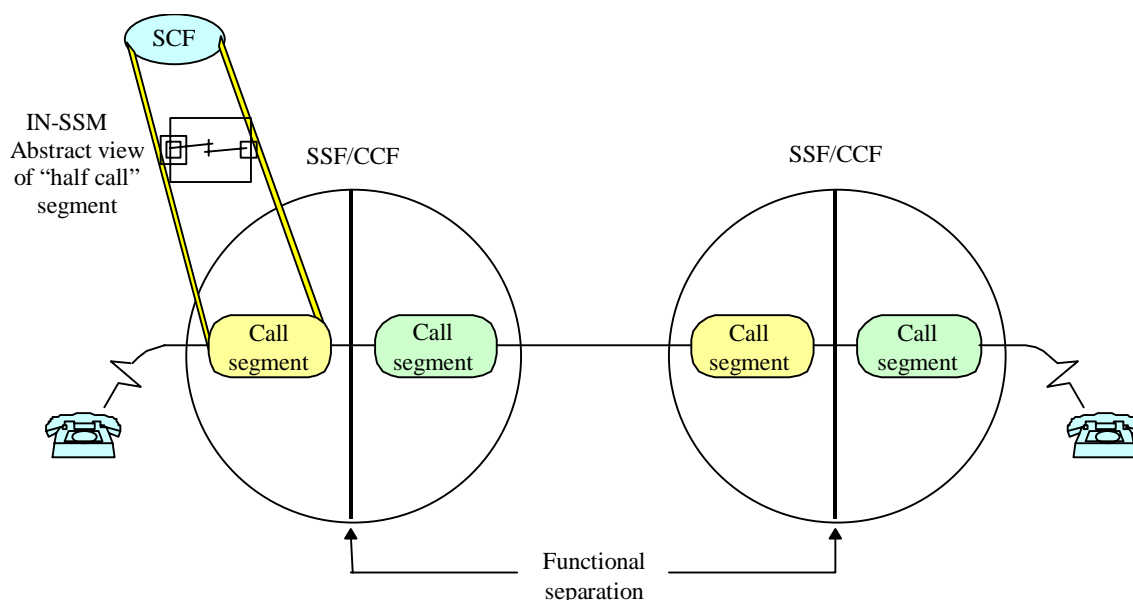
Figure 16: Call segments in a two-party inter-CCF/SSF call

This term "half-call" is used to refer to the physical resources (e.g. connectivity and transmission resources represented by legs and connection points) and to the processes (e.g. basic call processes as modelled by BCSMs) that are involved in the isolated portion of a call.

The SCF does not have direct access to a call segment, but rather has access to the abstract representation of the call segment provided by the Call Segment Association.
For IN, access via a Call Segment Association can be a single one-party, two-party or multi-party call segment, or to a pair of associated call segments (see figure 17). A set of associated call segments is 2 to n call segments that can be related together by the CCF/SSF and manipulated as a set (e.g. to merge them together into a single call segment). An example with 2 associated call segments is shown in figure 17. Each Call Segment being represented by a limited number of CSCV states defines the extent to which associated Call Segments are visible to the SCF via a CSA.

b) A Call Segment Association provides an SCF with an abstract view of a single two-party or multi-party call segment, or of a set of (1 to n) associated call segments. The CSA represents the properties of a call segment or pair of associated call segments of interest to the SCF (e.g. the connectivity and call processing aspects) and describes these properties in terms of objects (i.e. virtual resources) that can be manipulated by the SCF. For connection control, these objects include legs and connection points.

NOTE: An IN CS-1 connection control IN-SSM is identical to a CSA (Call Segment Association) in terms of connectivity context as defined here and described by the Connection View (CV) Model.
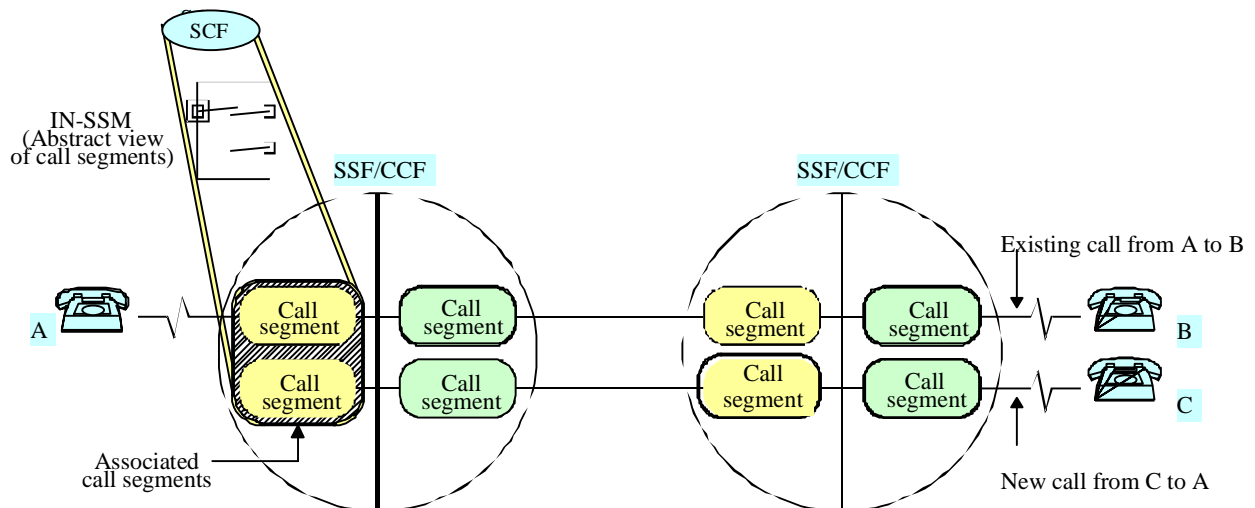
Figure 17: Associated call segments

The call segment concept can be used to describe how the definitions of "single-ended service feature" and "single point of control" apply to the distributed functional plane.

A *single-ended service feature* is described in terms of:

- The scope of control of the service logic instance that realizes the service feature, with respect to the call; and

- The interaction of the service logic instance with respect to other single-ended service logic instances on the same call.

The scope of control of a single-ended service logic instance is restricted to the isolated "half-call(s)" in a CCF/SSF (i.e. the call segments) accessible to the SCF via a relationship. This is illustrated in figure 18 for a two party call, which shows the BCSMs, related to each call segment.

This is also extended for a set of associated "half-calls", or a multi-party "half-call". These scenarios are illustrated in the subsequent figures 19 and 20.

All of these scenarios are based on the assumption that "half-calls" can be isolated from their complementary "half-calls" by the functional separation between an originating BCSM instance and its complementary terminating BCSM instance.

A single-ended service logic instance can only directly influence the processing of the isolated "half-call" (or associated "half-calls") in the CCF/SSF.
The other "half-calls" can only be indirectly influenced via information propagating from one "half-call" to another (i.e. between originating and terminating BCSMs, or between BCSMs in different CCF/SSFs).
As such, multiple single-ended service logic instances (one per "half-call") may be simultaneously active on a single call, each isolated from the other by the communication between "half-calls".
The communication between originating and terminating BCSMs in the same CCF/SSF is described in the clause of the BCSM description, and is illustrated by the "Basic primitive signal interface model". The communication between BCSMs in different CCF/SSFs is assumed to be the same as existing signalling between exchanges.

Single point of control, as it applies to the distributed functional plane is as follows:

a) an isolated "half-call" in the CCF/SSF can only be influenced by one SCF at a time;

b) while **one** SCF (controlling SCF) is influencing an isolated "half-call" in the CCF/SSF, it shall be possible to:

   - send DP reports from the CCF/SSF to the controlling SCF or different SCFs;

   - end the relationship between the controlling SCF and the CCF/SSF, then initiate a new relationship between the CCF/SSF and another controlling SCF (SLPI).

Multiple Point of Control, as it applies to the distributed functional plane is as follows:

  a) an isolated "half-call" in the CCF/SSF can be influenced by more than one SCF at a time;

  b) while one SCF is influencing an isolated "half call" in the CCF/SSF it shall be possible to:

   - send DP reports from CCF/SSF to the same SCF or different SCFs;

   - initiate a new additional relationship to a controlling SCF, or end a relationship between the controlling SCF and the CCF/SSF.
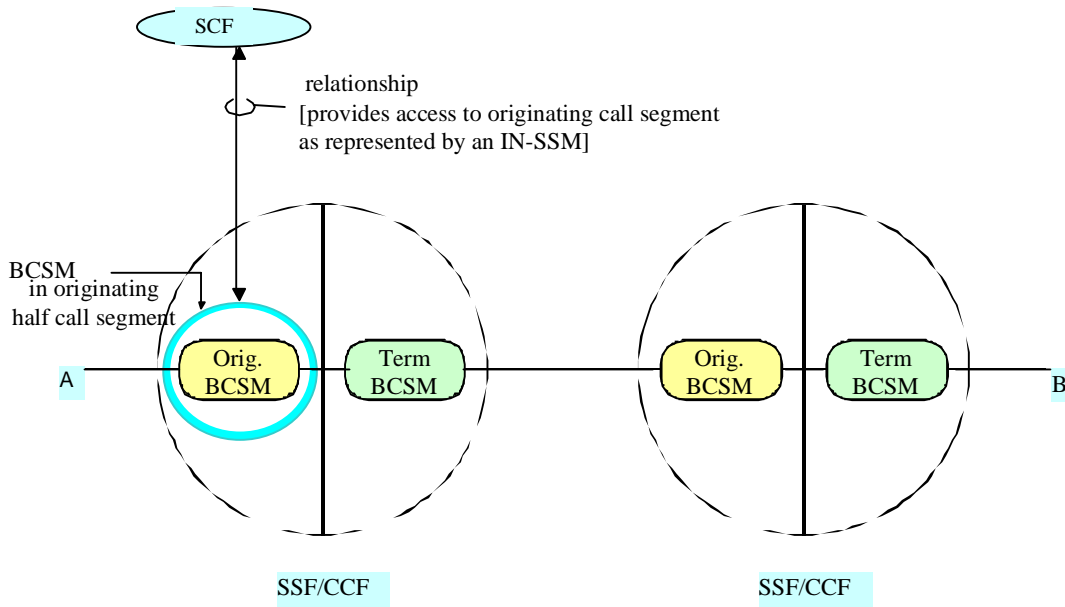
Figure 18: Single-ended control of a two-party call
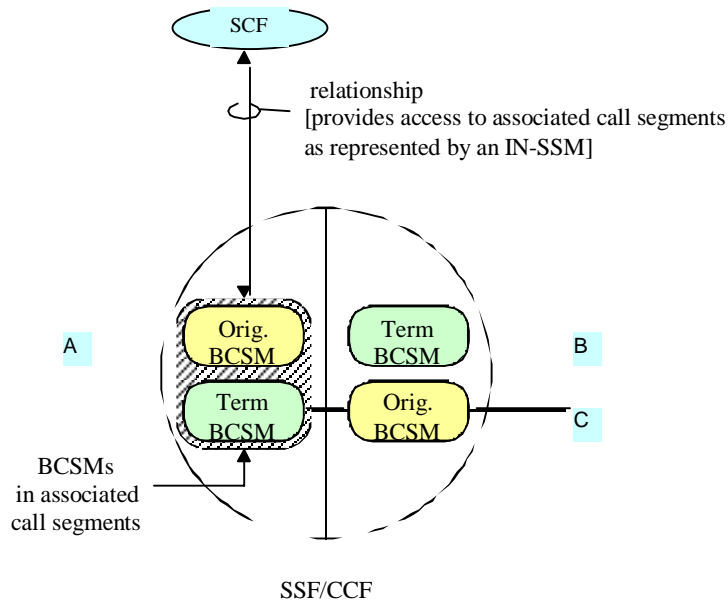
Figure 19: Single-ended control of associated calls
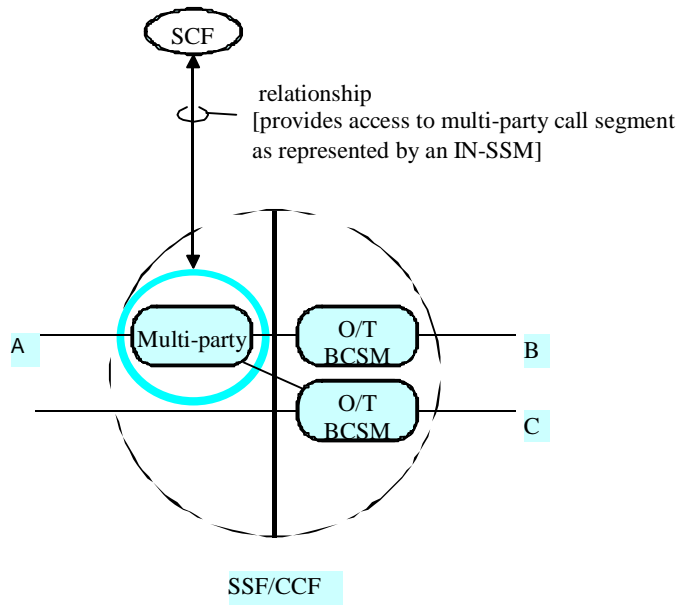
Figure 20: Single-ended control of a multi-party call

6.6.2   The Connection View Model

This clause describes the Connection View model, defines an inventory of CSA and CS Connection View states, and lists the allowable transitions between CSCVs.

The Connection View approach is based on the Connection View (CV) model. CV processing provides an SCF with the ability to influence existing call and connection processing capabilities. It does so by providing a generic representation of call and connection processing resources that support the processing capabilities of interest.

The CV processing within the CCF/SSF can be viewed as translating SCF instructions into operations that are understood by internal CCF/SSF call processing, as well as translating internal call processing events and the state of internal call processing resources into information which is understood by the SCF as indicated in figure 21.
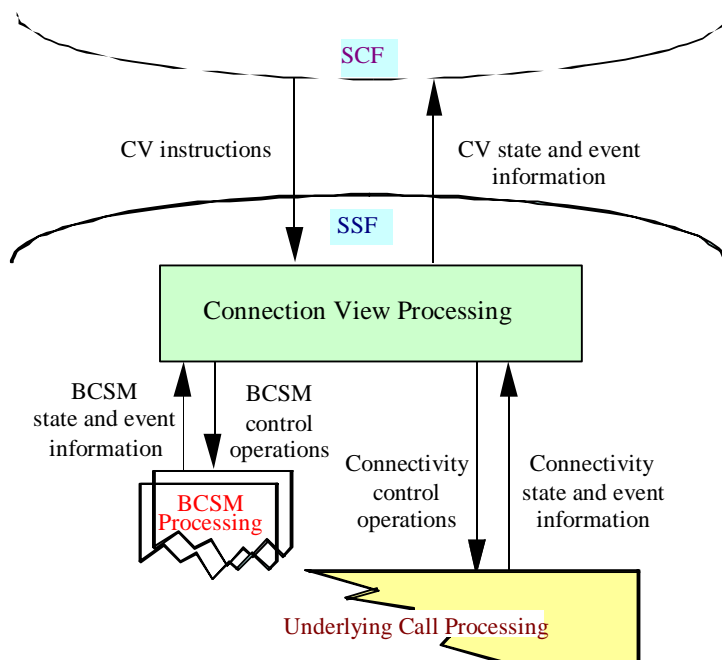
Figure 21: Connection View (CV) processing model

6.6.2.1   Connection View Objects

The CCF/SSF call processing and CV processing resources are described by a set of CV objects which include the following:

- Call Segment Association (CSA),

- Call Segment (CS),

- Legs,

- Connection Point (CP),

- BCSM.

The CV provides a view or abstraction of call and connection processing resources that is independent of supplier implementation and that only represents the essential characteristics of the resources needed by service logic, hiding the physical details and technical complexity of these resources. The CV reflects these properties in terms of *connectivity* context and a *call processing* context. The objects in the *call processing* context manage the relationship of a party to a basic two-party call and the objects in the *connectivity* context manage multiple two-party calls.

The *call processing* **context** reflects the state of the basic call processing required to set up and maintain the legs in a CS. Only one type of object is explicitly defined in the call processing context and that is BCSM. The BCSM represents the basic call processing required to establish and maintain a communication path from the CS toward an originating party and from the CS toward a terminating party.

The *connectivity* **context** reflects the state of a CS or a pair of associated CSs and includes the set of legs in the CSs and the relationship of each leg to a connection point (CP). The types of objects in the connectivity context correspond to legs, CPs, Call Segments (CSs), and pairs of associated CSs (CSA). Figure 22 provides the graphical examples of the object relationship in the connectivity context.
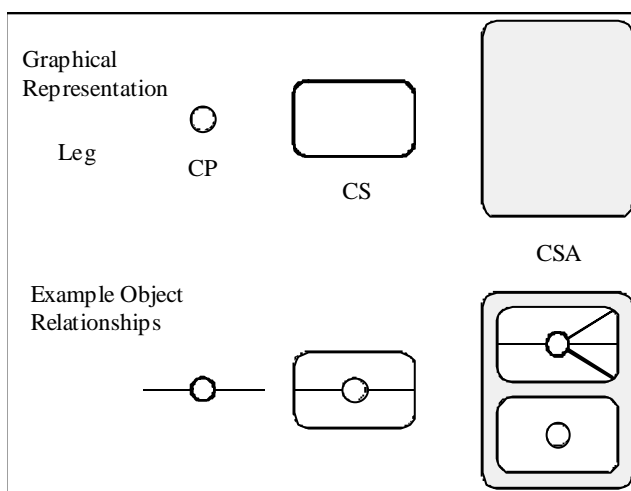
Figure 22: Connection View Objects

As indicated by figure 22, a Call Segment Association (CSA) object contains one or more Call Segments (CSs). Each CS object contains a single Connection Point (CP) object, which may be connected to leg objects. Depending on its status each of these objects has implicit or explicit attributes. These objects are further described below.

6.6.2.1.1   Call Segment Association Object

The CSA object contains one or more CSs that are associated in the context of IN control.

A Call Segment Association (CSA) instance is created when an IN service logic instance is invoked that requires IN connection control. It is either created as a result of encountering a TDP in a BCSM that satisfies DP criteria, or is initiated by the SCF independent of encountering TDPs.

A CSA instance is destroyed when the SCF informs the SSF that the IN service logic instance is completed or the CSA should be destroyed. The SSF can also initiate CSA destruction when an IN service relationship to the SCF is to be terminated (e.g. no pending reports or during error or abnormal conditions).

6.6.2.1.2 Call Segment Object

The Call Segment (CS) object contains a Connection Point and any attached legs.
IN service logic may request the manipulation of CSs via operations that act on the Connection View Objects. IN Service Triggering occurs in the "Initial Call Segment".

6.6.2.1.3 Leg Object

The leg object represents a communication path towards a real or virtual end user.

Leg: A representation within a call processing state model representing a telecommunication path towards some addressable entity e.g. a path toward a user, intelligent peripheral unit etc. (ITU-T Recommendation Q.1290).

A *leg* can be designated as a controlling leg or as a passive leg.

- The **controlling leg** is the leg that represents the local access interface at local exchange or the remote access interface at transit exchange (e.g. the incoming line or trunk in an originating call segment, or the outgoing line or trunk in a terminating call Segment). It is the leg for which IN service logic program instances are invoked, either as a result of end user signalling (e.g. a mid-call event) or on behalf of an end user (e.g. individual-based triggering on a line or SCF initiated call). There is no more than one joined or pending controlling leg in a CSA. Transfer of ownership from an end user supported by a controlling leg to an end user supported by a passive leg is not feasible (see note).

- The **passive leg(s)** are directed towards the other half-call.
  Any other leg in a Call Segment Association than the controlling leg.

  NOTE: The controlling leg represent line or trunk interfaces. The example of Connection View state transitions given presents limitations on manipulation of these two types of controlling legs. Especially, "third-party" control aspect which has capability of initiating call/connection set up between two parties from the "third-party" side is supported partially by defining controlling leg status as "surrogate".

The following four **status values** are specified for the leg:

- **Joined,** indicating that the leg is joined to the connection point and exists.

- **Pending**, indicating that the leg is in a call set-up state (i.e. the call is not yet stable) or a call clearing phase.

- **Surrogate** (for a controlling leg), indicating that the leg supports a communication path towards a virtual party (e.g. the network, or a party that forwarded the call), rather than an external end party. The "surrogate" leg status signifies participation of a "third- party" in the call, but not in the connection. It may also represent a relationship with the controlling user where the remote party has been on hold and a possible charging ownership.

Legs are uniquely identifiable for the SLPI in an IN Call Segment Association.

For IN it is possible:

a) To influence the flow of basic call processing associated with a leg (e.g. generate a signalling event and continue basic call processing as appropriate for that event).

b) To add a passive leg to a Call Segment Association by originating a call or terminating a call; to drop legs (one or more) by clearing calls.

c) To make or break connections between legs (e.g. join or split).

d) To move legs from one connection point to another within the same Call Segment Association (e.g. split a leg from one connection point then joining it to another). It is possible to move a leg from one call segment to another call segment within the same CSA or to move a leg from one CSA to another.

6.6.2.1.4   Connection Point Object

A *connection point* represents a joint function between two legs, a conference function between three or more legs, replication function, merging function, or an information distribution function between two or more legs that specifies the directionality of information flow through the connection point (e.g. the connection point could receive information from multiple legs and/or from a SRF resource and distribute it to another leg). It interconnects legs supported by equivalent bearer services, and supports interworking between circuit mode/speech and circuit mode/3,1 kHz audio bearer services.

There can be 1 to n connection points in a Call Segment Association. Each connection point is related to a Call Segment that is represented by the CSA. In a Call Segment Association, it is possible to merge two connection points into a single connection point, thereby merging the corresponding call segments. Finally, it is possible to release a connection point and all of its legs all at once, thereby clearing the corresponding Call Segment.

6.6.2.2   Relationship of BCSM to Connection View CS States

Figure 23 shows, at a high level, the relationship of the Connectivity Objects that make up the Connection View CS States to the Call Processing Objects (BCSMs).
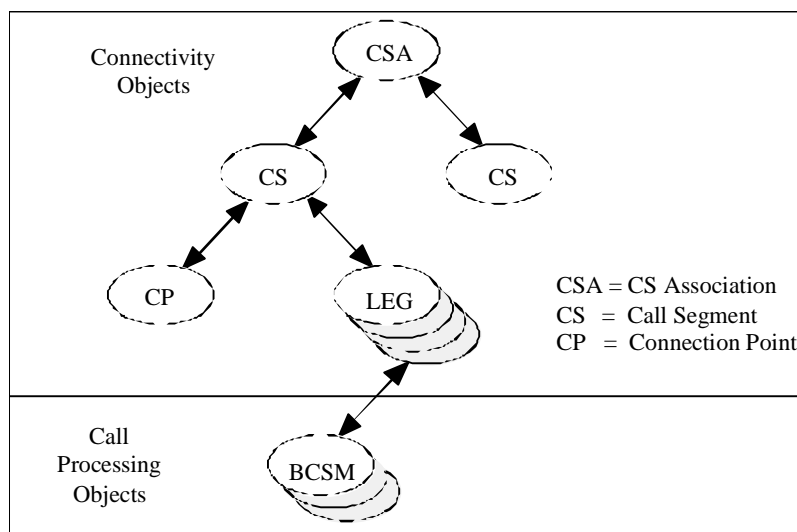
Figure 23: Connection View CS State objects and their relationship with BCSM

A Connection View CS State shows the connectivity between a Controlling Leg and one or more Passive Legs (see below for definitions of terms).

A single BCSM model the Originating or Terminating call processing to set up and maintain a two-party call.
There is a single instance of a BCSM for each Passive Leg of a Connection View CS State (CSCV).

In case no passive leg is present within a CS in the CSCV, the BCSM instance belongs to the controlling leg in that CS.

A Call Segment with only a controlling leg (status "joined") left is assigned a BCSM instance (Call processing Context) to supervise the leg. The "BCSM type" attribute set on the controlling leg is used to reflect the type of BCSM (originating or terminating) to be assigned. The corresponding FSM for the Call Segment is put into the "Waiting_For_Instruction" state when a leg is left in a CS on request from the SCF due to a CPH operation
(e.g. DisconnectLeg). The call processing is suspended (e.g. at the O/T_MidCall DP) in the associated BCSM instance.

When the controlling leg becomes connected again to another passive leg (e.g. MergeCallSegments, Connect operation) and the BCSM instances are the same type no BCSM instance will be connected to the controlling leg anymore. The DPs armed and other pending reports for the controlling BCSM instance will be transferred to the passive leg BCSM instance.

6.6.3   Connection View State Transitions

6.6.3.1   Introduction

From a protocol perspective the Connection View objects have been combined to create a set of standardized CSA Connection Views (CSACVs) and a set of standardized CS Connection Views (CSCVs). The protocol controls the CSAs and the CSs respectively represented by the CSA and CS Connection View states. The CSs representing the same or different CSCV states may be associated in a CSA object described by appropriate CSACV states. Note that the number of passive legs in a CS and the number of CSs into a CSA are not limited to two, allowing to model all stable multiparty calls with three or more parties.

The SSME-Control functionality is extended in order to control the creation of the CSA instances and to provide an overall co-ordination function for managing the procedures who have an impact on more than one CSA.

Next the SSME-Control procedures and transitions are described followed by a description of the CSA procedures and transitions- Hereafter, the CS procedures and transitions are detailed.

6.6.3.2   SSME-Control

6.6.3.2.1   Introduction

The SSME-Control functionality is extended in order to control the creation of the CSA instances and to provide an overall co-ordination function for managing the procedures who have an impact on more than one CSA (e.g. MoveCallSegments).

A CSA instance is created:

- on receipt of a CreateCallSegmentAssociation operation;

- on receipt of an InitiateCallAttempt operation as the first operation of an SCF-initiated dialogue;

- on determination that a service triggering shall occur.

6.6.3.2.2   Transition table for SSME-Control

The SSME-Control transitions related to the CPH Handling are described in table 15.

The signals used in the transition table for SSME-Control are as follows:

- The ExportCSReq signal shall request the addressed CSA process instance to export the indicated call segment.

- The ExportCSResp signal shall be returned by the addressed CSA process instance to indicate to the SSME-Control that the export of the call segment was successfully executed.

- The ImportCSReq signal shall request the addressed CSA process instance to import the indicated call segment, since unique numbering of the legs are required on a CSA basis renumbering of the legs for all remaining CSs are required.

- The ImportCSResp signal shall be returned by the addressed CSA process instance to indicate to the SSME-Control that the import of the call segment was successfully executed.

Table 15: SSME-Control Transition Table

Event	Action
Triggering received from FIM	Create an instance of a CSA process (csa1 signal), allocate a CSAID, update the routing table and associate the signalling controller CallRef with the newly created CSA.
InitiateCallAttempt	Create an instance of a CSA process (csa1 signal), allocate a CSAID and update the routing table. NOTE: For CS-1 the InitiateCallAttempt always creates a new CSA process instance, for CS-2 onwards the ICA may be proceeded by a CreateCallSegmentAssociation, in which case the InitiateCallAttempt shall not create a new CSA.
CreateCallSegmentAssociation	Create an instance of a CSA process (csa1 signal), allocate a CSAID, update the routing table, sent a CreateCallSegmentAssociationResult to SCF
MoveCallSegments	Send an **ExportCSReq** event in order to request the CSA to export the source CS and perform the leg numbering by the CSA prior to exporting the source CS and wait for **ExportCSResp**. After receiving the **ExportCSResp** sent an **ImportCSReq** in order to import the previously exported CS into the target CSA with the identifier of the newCallSegment and wait for **ImportCSResp**. After receiving the **ImportCsResp** send a MoveCallSegmentsResult to the SCF.

6.6.3.3 Call Segment Association Connection View (CSACV) Transitions

6.6.3.3.1 Introduction

This clause describes from a protocol perspective the Call Segment Association Connection View (CSACV) processing for each state, and gives examples of the events or the SCF operations that result in creating a new CSACV state, destroying a CSACV state, and transitioning from the one CSACV state into another CSACV state. The transitions between CSACV states occur due to end user actions (e.g. off-hook or disconnect), switch processing (e.g. switch-based features), or as a result of processing SCF operations.

6.6.3.3.2 Functional Procedures for Call Segment Association (CSA)

6.6.3.3.2.1 Queuing of BCSM events and operations in the CSACV

In the case when a RequestReportBCSMEvent (RRB), ApplyCharging. FurnishChargingInformation, RequestNotificationChargingEvent or CallInformationRequest operation is received by the CSACV with legID for which no corresponding passive leg yet exist, the "to be armed" BCSM events respective ApplyCharging. FurnishChargingInformation, RequestNotificationChargingEvent or Call InformationRequest shall be queued at the CSA level.

Only RRB arming requests respective ApplyCharging. FurnishChargingInformation, RequestNotificationChargingEvent or Call InformationRequest for one new LegID ("leg to be created") should be accepted and queued.
The EDP arming requests may be provided in one or more RRB operations from the SCF. It is the task of the SSF to accumulate all RRB arming requests to be queued for the same LegID.
When an operation (e.g. Connect) subsequently creates a new passive leg with a legID equal to a legID for which BCSM events respective ApplyCharging, FurnishChargingInformation, RequestNotificationChargingEvent and/or Call InformationRequest operations have been queued, the queued BCSM events and queued operations shall be retrieved and transferred to the exiting CS instance.

Receiving a RRB for arming BCSM events respective an ApplyCharging. FurnishChargingInformation, RequestNotificationChargingEvent or Call InformationRequest operation on an unknown legID while RRB arming requests are already queued for a different legID should be processed as an error "Unknown legId".

A RRB for disarming at least one DP on an unknown legid is processed as en error "Unknown legId".

The queued RRB arming requests respective queued ApplyCharging. FurnishChargingInformation, RequestNotificationChargingEvent and Call InformationRequest operations are maintained at the CSA level. They disappear when the CSA is deleted or when the new leg is created by an operation referencing the corresponding legID.

6.6.3.3.3   Transition diagram for Call Segment Association (CSA)

The transition diagram for the CSA is given in figure 24.

> NOTE: Figure 24 shows examples and does *not* show all possible transitions.
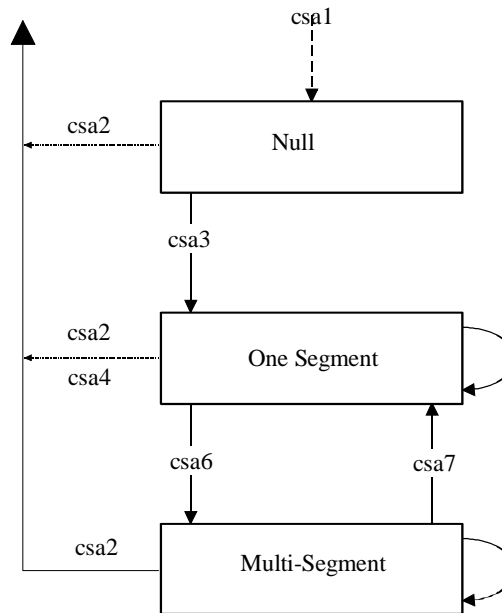
Figure 24: CSACV transition diagram

The states identified for the CSACV transition diagram is:

Null: The state Null represents the condition when a call segment association is created without any call segments.

One Segment: This state represents a call segment association containing one call segment.

Multi-Segment: This state represents a call segment association containing multiple call.

The CSACV transition diagram contains the following transitions:

csa1:   Create Call Segment Association
csa2:   Delete Call segment Association(ReleaseCall)
csa3:   Triggering event from FIM (examples are: SetupInd, SetupReqInd), InitiateCallAttempt
csa4:   ReleaseCall
csa5:   DisconnectLeg
csa6:   SplitLeg, InitiateCallAttempt
csa7:   ReleaseCall, MergeCallSegments
csa8:   SplitLeg, InitiateCallAttempt, MergeCallSegments, DisconnectLeg, MoveLeg

6.6.3.3.4   Transition table for Call Segment Association (CSA)

The Call Segment Association (CSA) process type transitions related to the CPH Handling are described in table 16.

The signals used (see also SDL Diagrams) in transition table for Call Segment Association (CSA) are as follows.

The ExportCSReq signal shall request the addressed CSA process instance to export the indicated call segment.

The ExportCSResp signal shall be returned by the addressed CSA process instance to indicate to the SSME-Control that the export of the call segment was successfully executed.

The ImportCSReq signal shall request the addressed CSA process instance to import the indicated call segment, since unique numbering of the legs are required on a CSA basis renumbering of the legs for all remaining CSs are required.

The ImportCSResp signal shall be returned by the addressed CSA process instance to indicate to the SSME-Control that the import of the call segment was successfully executed.

The SetLegLocation signal shall set the location of a leg, i.e. to which call segment a given leg identifier given in the signal is belonging to.

The GetLegLocation signal shall request the location of a leg, i.e. the call segment identifier to which the leg identifier given in the signal is belonging to.

The ImportLeg signal shall invoke to move a communication path in a CS (in the process of moving the path from another CS to this CS). This operation shall add the existing leg specified by the leg identifier to the CS. For a passive leg the number of legs within a CS are incremented by one and the leg vector is appended. For the controlling leg the leg status shall change from "shared" to "joined".

The ExportLeg signal shall remove the leg specified by the leg identifier to the specified CS, but shall not destroy the leg. For a passive leg the number of legs shall be decremented by one, and the state vector is removed. For the controlling leg, the leg status shall change to "shared".

The RemoveLeg signal shall remove the leg from the CS that identifier specifies. The number of legs in a CS is decrements by one and the associated state vector is removed. The path corresponding to the removed leg is released as a result of this operation.

Table 16: Call Segment Association (CSACV)Transition Table

CSACV State ⇒ Event ⇓	Null	One Segment	Multi Segment
SplitLeg operation	- Discard operation and return error (Task Refused) - Remain in **same state**	- Check SplitLeg Argument - ExportLeg from currently residing segment - Create Call Segment process instance - ImportLeg into new call segment - Pass: Splitleg(c) to the new call segment or SplitLeg(p) to the segment where the leg is exported from - Go to **Multi Segment**	- Check SplitLeg Argument - ExportLeg from currently residing segment - Create Call Segment process instance - ImportLeg into new call segment - Pass: Splitleg(c) to the new call segment or SplitLeg(p) to the segment where the leg is exported from - Remain in **Multi Segment**
Initiate-Call-Attempt operation	- Check InitiateCallAttempt Argument - Create Initial Call Segment process instance - SetLegLocation for the passive and controlling legs to the newly created CS - Pass InitiateCallAttempt to the newly created CS - Go to **One Segment**	- Check InitiateCallAttempt Argument - Create Call Segment process instance - SetLegLocation for controlling and passive legs to the newly created CS - Pass InitiateCallAttempt to the newly created CS - Go to **Multi Segment**	- Check InitiateCallAttempt Argument - Create Call Segment process instance - SetLegLocation for controlling and passive legs to the newly created CS - Pass InitiateCallAttempt to the newly created CS - Remain in **Multi Segment**
Merge-Call-Segments operation	- Discard operation and return error (Task Refused) - Remain in **same state**	- Discard operation and return error (Task Refused) - Remain in **same state**	- Check MergeCallSegments argument - Move all legs from the source CS to the target CS - Delete the source CS - Remain in **Multi Segment**, if number of CSs is > 1 otherwise go to **One Segment**

CSACV State ⇒ Event ⇓	Null	One Segment	Multi Segment
Disconnect-Leg operation	- Discard operation and return error (Task Refused) - Remain in same **state**	- Check DisconnectLeg Argument - GetLegLocation to locate the CS where the leg shall be removed - RemoveLeg in CS - Pass DisconnectLeg to the CS where disconnection shall occur. - Remain in **same state** if number of legs left in call segment left is greater than zero else move to **Null**	- Check DisconnectLeg Argument - GetLegLocation to locate the CS where the leg shall be removed - RemoveLeg in CS - Pass DisconnectLeg to the CS where disconnection shall occur. - Remain in **same state** if number of legs left in call segment is greater than zero or number of call segments left is greater than 1 else move to **One_Segment**
MoveLeg operation	- Discard operation and return error (Task Refused) - Remain in **same state**	- Discard operation and return error (Task Refused) - Remain in **same state**	- Check MoveLeg Argument - ExportLeg from the CS where it is currently residing. - ImportLeg into the target CS - Remain in **same state** or move to the **One_Segment** state if it is the ExportLeg of the last passive leg of a CS
Import-CSReq	- Check ImportCSReq – Import CS and assign the CS identifier - Send ImportCSResp to SSME-Control - Go to **One Segment**	- Check ImportCSReq - Import CS and assign the CS identifier - Send ImportCSResp to SSME-Control - Go to **Multi Segment**	- Check ImportCSReq - Import CS and assign the CS identifier - Send ImportCSResp to SSME-Control - Remain in **same state**
Export-CSReq	- Discard operation and return error (Task Refused) - Remain in **same state**	- Check ExportCSReq - Export CS the given source CS - Send ExportCSResp to SSME-Control - Go to **Null**	- Check ExportCSReq - Export CS the given source CS When the CS with the controlling leg joined is exported the status of the controlling leg in all the other CSs are set to surrogate. (This is done by the ExportControllingLeg procedure) - Send ExportCSResp to SSME-Control - Remain in same **state** if number of call segments left is greater than 1 else move **to One_Segment**

6.6.3.4 Call Segment Connection View (CSCV) Transitions

6.6.3.4.1 Introduction

This clause describes from a protocol perspective the Call Segment Connection View (CSCV) processing for each state, and gives examples of the events or the SCF operations that result in creating a new CSCV state, destroying a CSCV state, and transitioning from the one CSCV state into another CSCV state. The transitions between CSCV states occur due to end user actions (e.g. off-hook or disconnect), switch processing (e.g. switch-based features), or as a result of processing SCF operations. Note that the transition events related to user interactions (e.g. ConnectToResource) are outside the scope of this capability set.

Table 17 followed by CSCV state transition figures at the end of this clause provide a concise summary and overview of these transition events.

The transition figures are not intended to cover all possible transitions but merely to provide an overview.

IN-initiated transitions may be categorized according to the manner in which they affect the CSCV state:

1) *BCSM-only transitions*: Occur when the BCSM changes state (or continues normal processing) with no explicit change to the CSCV state. For example, if SSF sends an InitialDP from the "Originating_Setup" CSCV state, and SCF responds with Connect, the BCSM changes state, but no explicit change to the initial CSCV state occurs. Note that, in processing the BCSM request, there *may* be an implicit change to the CSCV state through non-IN processing (e.g. a switch-based screening feature).

2) *SCF-requested connectivity changes:* Occur when SCF explicitly requests a change in connectivity. This request may or may not cause a CSCV state transition.

Within this clause, each CSCV state is described with respect to the following:

a) ***Relationship with the BCSM***: a list of the Points in Call (PICs) and DPs that are associated with this CSCV state. Note that the PICs are associated with the passive legs (i.e. *Leg* p1 or *Leg* p2) only, to avoid the ambiguity that would otherwise result for the "Stable_Multy_Party", "Stable_Multi_Passive_Party", and "Forward" CSCV states. Additional detail on the mapping of DPs and events to the CSCV states is provided in this clause.

b) ***Entry Events***: the events that cause the call processing and CV processing associated with this CSCV state to be initiated.

c) ***Exit Events***: the events that signify normal completion of the call processing and CV processing associated with this CSCV state. This item begins with a summary of the SSF events (i.e. TDPs, EDPs, and BCSM events) relevant to the CSCV state approach to Call Party Handling. A list of SCF responses that cause CSCV state transitions follow.
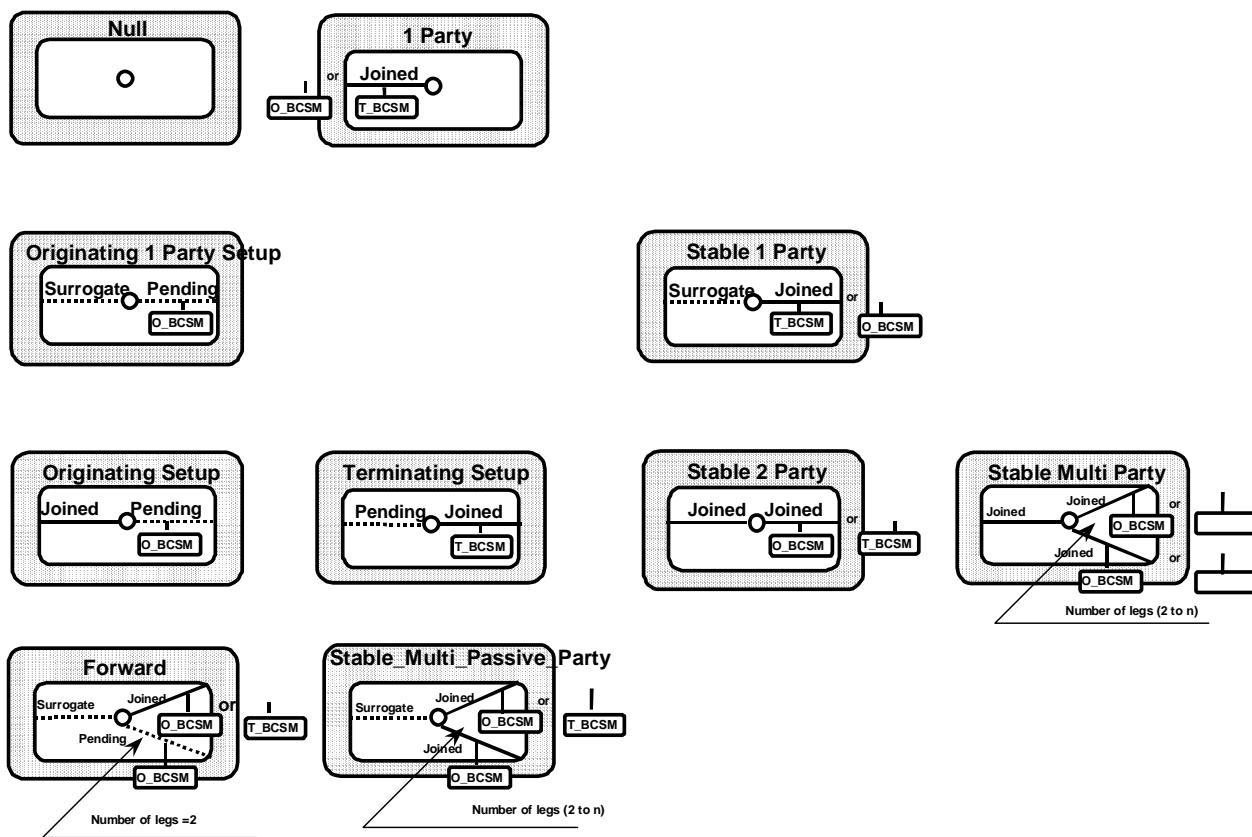
Figure 25: Call Segment Connection View (CSCV) States

Figure 25 depicts the CSCV states to be supported by the CV processing.

In subsequent clauses detailed descriptions of these Connection View States (CSCV states), including their relationship with the BCSM and the transition events between CSCV states are provided.

CSCV State Definitions:

   Null:

   This state represents the condition where no call processing is active and there is no controlling leg or passive leg connected to the connection point.

   Originating_Setup:

   This state represents a call segment instance with a joined controlling leg and a pending passive leg with an associated O_BCSM (e.g. an originating two-party call in the setup phase).

Terminating_Setup:

>This state represents a call segment instance with a pending controlling leg and a joined passive leg with an associated T_BCSM (e.g. a terminating two-party call in the setup phase).

Stable_2_Party:

>This state represents a stable two-party call, and is either an originating or a terminating call from the perspective of the controlling user with a joined controlling leg and a "joined" passive leg with either an O_BCSM or a T_BCSM associated with it.

1-Party:

>This state represents a 1-party call with a joined controlling leg status and no passive legs. The O_BCSM or T_BCSM is associated with the controlling leg since no passive leg is connected to the CP.

Originating_1_Party Setup:

>This state represents a 1-party call with a surrogate controlling leg and a pending passive leg with which an O_BCSM is associated.

Stable_1_Party:

>This state represents a 1-party call with a surrogate controlling leg and a joined passive leg with either an O_BCSM or T_BCSM associated with it, that is in a stable phase.

Forward:

>This state represents a forwarded call with a surrogate controlling leg, a joined passive leg with which an O_BCSM or a T_BCSM is associated, and a pending passive leg with which an O_BCSM is associated.

Stable_Multi_Passive_Party:

>This state represents a call with a surrogate controlling leg and two to N joined passive legs with each of which an O_BCSM or a T_BCSM is associated. The call between the involved passive legs is in the stable phase.

Stable_Multi_Party:

>This state represents a multi-party call with a joined controlling leg and two to N joined passive legs with each of which an O_BCSM or a T_BCSM is associated.

6.6.3.4.2 Functional Procedures for Call Segment (CS)

6.6.3.4.2.1 BCSM type indication

A "BCSM type" attribute shall be allocated with the controlling leg at the moment this leg is created, indicating if the controlling leg belongs to an Originating or a T-BCSM, i.e. an indication for the type of BCSM in which call triggering occurred.

When only the controlling leg is connected to the Connection Point in a CS, the "BCSM type" attribute shall determine which BCSM shall apply. The conditions of the leg i.e. the armed EDPs, the ApplyChargingReport pending, the EventNotificationCharging pending, and the CallInformationReport pending are also applied to the same leg after creation of an BCSM instance (e.g. after a SplitLeg). The latter entails that if no EDPs and/or reports were armed for the leg in relation to the created BCSM type, then an explicit arming is to be made using e.g. the RequestReportBCSMEvent operation, if needed by the SLP.

In order to obtain full alignment with the SLP design in the SCF no mapping of events from O_BCSM to T_BCSM events and vice versa shall occur. This means that the DP at which call processing is to be suspended shall be selected in the associated BCSM instance based on the actual state of the call as seen from the legs viewpoint; e.g. if in an "Active PIC" then the BCSM instance shall be put in MidCall DP of the active BCSM state when the leg is left in a CS after e.g. a DisconnectLeg or SplitLeg operation.

The following is noted for the O_BCSM and T_BCSM instances:

When the O_BCSM instance is put in the O_Mid_Call DP then:

- a transition to the Analyse_Information or Select_Route is possible by means of sending the appropriate operations from the SCF. This allows to support follow-on calls in accordance with the extended BCSM transitions as described herein.

When the T_BCSM instance is put in the T_Mid_Call DP then:

- in order to allow to set-up an originating call connection to another terminating called party the ContinueWithArgument followed by the InitiateCallAttempt and the MergeCallSegments operations may be sent from the SCF in order to perform a "transfer call" (see figure 26). It is noted that the transfer of a call in this case where the controlling leg has the T-BCSM type attribute is only possible in a serving node exchange (not in a transit exchange).
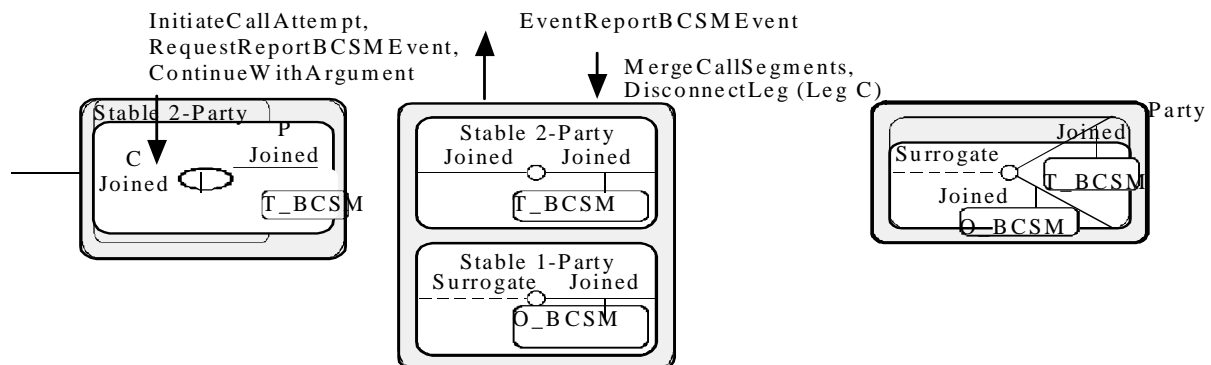
Figure 26: Example Transfer Call at T_BCSM

If in the above-mentioned cases, the controlling joined leg is the only leg left in a CS due to a SCF operation (DisconnectLeg, SplitLeg, MoveLeg, etc.), the CS shall be put in the 1-Party state with the associated SSF_FSM state in "Waiting For Instructions". In order to resume from the O_Mid_Call and T_Mid_Call DPs of the associated BCSM states a ContinueWithArgument operation can be sent by the SLPI.

It should be noted that the problem of stranding a "surrogate" controlling leg does not exist as it does for a "joined" controlling leg, because it is assumed that a stranded surrogate leg will be destroyed due to the fact that no voice path and no signalling is associated with this controlling leg.

6.6.3.4.2.2   Rules for inter-BCSM precedence Event handling rules

For Call Party Handling, it is possible to detect the same event (e.g. mid-call or user abandon) on multiple BCSMs. However, it is necessary to provide only one report of the given event to the SLPI in the SCF.

6.6.3.4.2.2.1   Event Detection Rules

The inter-BCSM precedence rules for the *detection* **of events** in CCF/SSF signalled on the **controlling leg** follow: Two cases applies:

1. The event signalled on the controlling leg does *not* indicate the particular passive leg to which it applies (e.g. analog hook-flash), then the following event detection rules apply:

    a) If the CSA contains multiple call segments (CSs), the event signalled on the controlling leg should be detected only on the CS with a controlling *legStatus* of "joined" or "pending".

    b) If the CS to which the event applies is a multi-party CS, then the event should be detected on all BCSMs within this multi-party CS.

2. The event signalled on the controlling leg indicates the particular passive leg to which it applies (e.g. a party identifier indication), then the mid-call event should be detected on the BCSM associated with the indicated passive leg. Any support for this feature is outside the scope of this capability set.

However, in all cases the event is reported only once to the SCF according to the Event Reporting Rules defined below.

6.6.3.4.2.2.2   Event Reporting Rules

The following inter-BCSM precedence rules for the **reporting to the SCF of events** to be signalled for the controlling leg shall apply:

1.   An event shall only be reported once to the SCF irrelevant of the number of passive legs per CS connected to the controlling leg (via CP).

2.   In case more than one event is possible due to different BCSM states prevailing for the different passive legs, then the reported event on the controlling leg shall be the event reflecting the most advanced call state of the BCSMs. For example in case of a CSCV Multi Party call with two passive legs, one passive leg (O_BCSM) in alerting (before answer) or release state(after answer) and the other passive leg (O_BCSM) in Active Call PIC. When calling party goes on-hook and a release messsage is received from calling party, two events are possible to report on the controlling leg, either O-Abandon or O-Disconnect. In this case O-Disconnect is reported according to the most advanced call state of the BCSMs, which is PIC Active call state.

3.   A "BCSM type" attribute on the controlling leg indicates if the controlling leg belongs to an Originating or a Terminating -BCSM. The "BCSM type" attribute is set to reflect the type of BCSM (originating or terminating) in which the trigger or event occurred at the moment the CS (i.e. initial CS in the CSA) was created.

  -   If the controlling leg belongs to an O-BCSM it is reported as an originating event.

  -   If the controlling leg belongs to a T-BCSM it is reported as a terminating event.

6.6.3.4.2.3   Creation of O/T_BCSM based on "BCSM type" attribute

A CS with only a controlling leg (leg status "joined") left is assigned a BCSM instance to supervise the leg. The "BCSM type" attribute set on the controlling leg is used to reflect the type of BCSM (originating or terminating) to be assigned. The corresponding SSF_FSM for the CS shall be put in "Waiting_For_Instructions" state where a leg is left in a CS due to a CPH (e.g. DisconnectLeg) operation and call processing is suspended at the Mid Call DP in the associated BCSM.

When the controlling leg becomes connected to the passive leg, no BCSM instance shall be connected to the controlling leg provided the same type of BCSM applies for controlling and passive leg. The BCSM instance which was connected to the controlling leg disappears in case a passive leg is moved (imported) to the Call Segment (e.g. MergeCallSegments operation). If a new passive leg is created (e.g. Connect operation) within the CS then the existing BCSM instance becomes connected to the passive leg.

6.6.3.4.2.4   Release of a Call/Connection

Release Event Processing General Rules:

  a)   When a DP related to a release event (e.g. *O/T_Disconnect, O_CalledPartyBusy, T_Busy, O/T_NoAnswer, Route_Select_Failure)* on a leg is reported to the SCF the LegId value and its pointer to a BCSM instance are retained until call processing at the current DP is resumed.

  b)   If the release occurs on a joined leg in the CS, the leg state is changed as follows: c-leg: "joined" to "surrogate"; p-leg: "joined" to "pending".

  c)   If the release occurs on the last joined leg in the CS, the leg state is changed without changing the CSCV state and the CS is deleted upon resumption of the call processing.

  NOTE: This implies *transient existence* of e.g. CSCV state "Originating_Setup" and "1_Party" with a surrogate controlling leg and "Terminating_Setup" with two pending legs.

  d)   Resumption (using Continue, or ContinueWithArgument) following a release event reported for *a Controlling Leg*, causes propagation of the release event, the release of passive legs in the Call Segment, and deletion of the Call Segment.
      Exception: Resumption when two or more than two joined passive legs are connected in the CS before resumption does not result in the propagation of the release event, all passive legs are retained and the CS continues.

  e)   Resumption (using Continue, or ContinueWithArgument) following a release event reported for *a Passive Leg*, causes removal of the passive leg from the Call Segment and the propagation of the release event.
      Exception 1: When more than two joined legs are connected in the CS, the resumption causes removal of the passive leg but does not result in the propagation of the release event.

A few examples are given below to illustrate the actions performed at release of a call and connection.

EXAMPLE 1:   The release of a Call/Connection from A-Party for a normal call where both call segments are in the "Stable_2_Party" state (with the associated O_BCSM and T_BCSM in **the active state**) is given in figure 27.
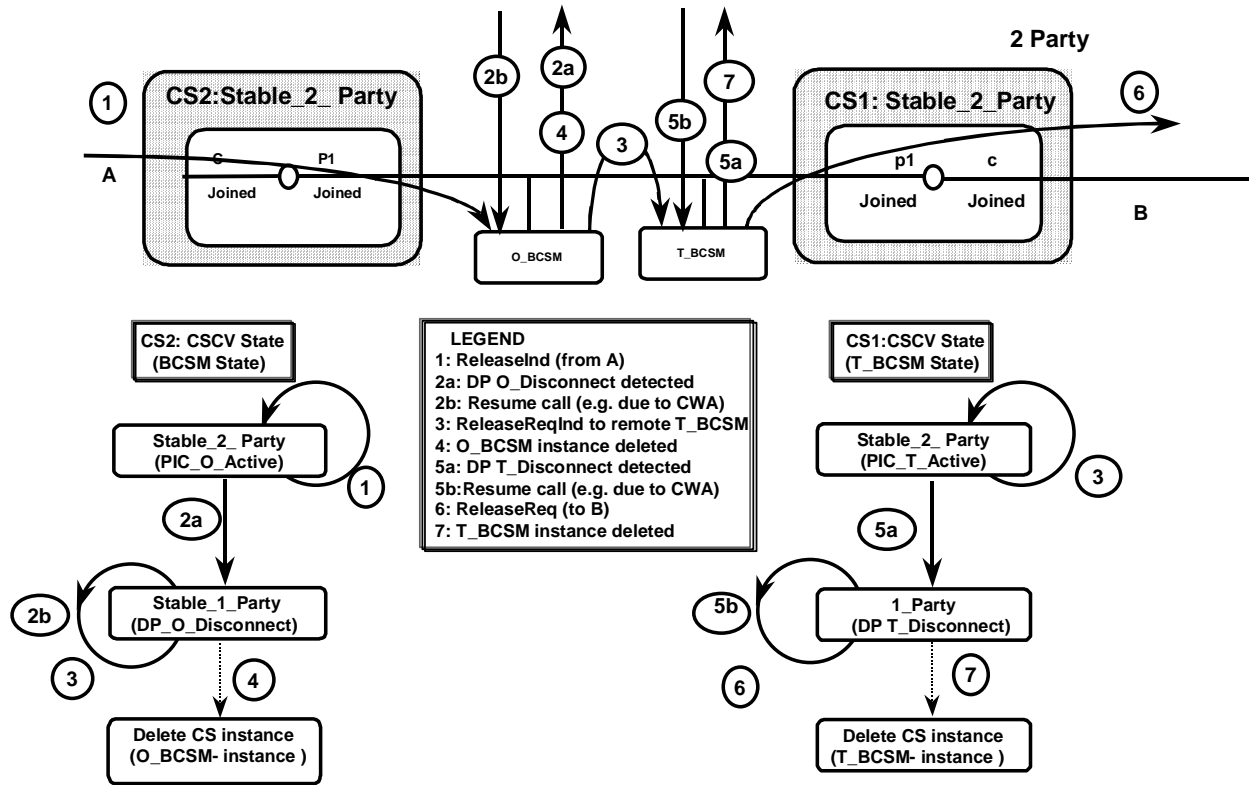
Figure 27: Example 1: Release from A-Party of Call/Connection where call segments are both in "Stable_2_Party"

EXAMPLE 2:   The release of a Call/Connection from B-Party for a normal call where both call segments are in the
               "Stable_2_Party" state (with the associated O_BCSM and T_BCSM in the active state) is given in
               figure 28.
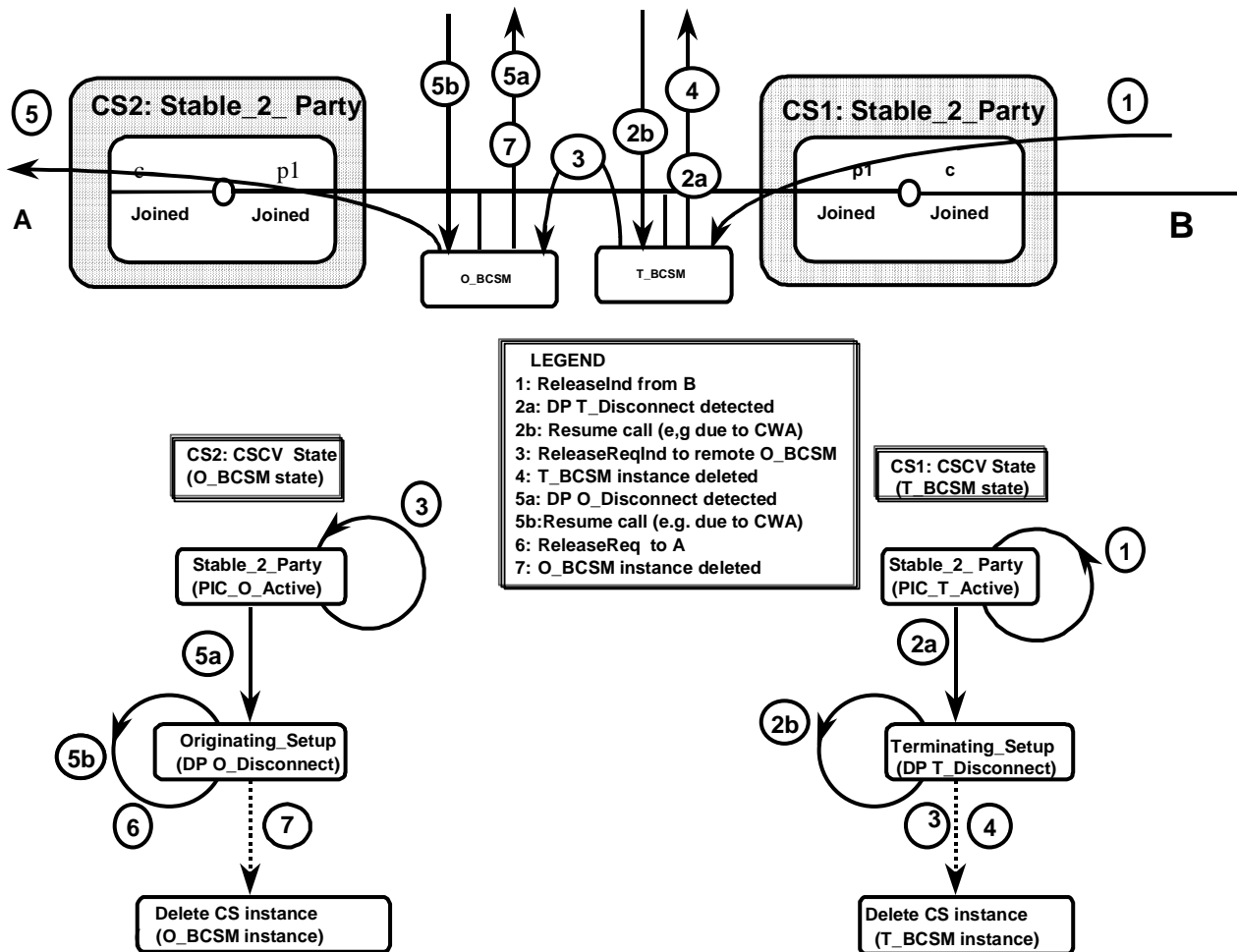
**Figure 28: Example 2: Release from B-Party of Call/Connection where call segments are both in
"Stable_2_Party"**

EXAMPLE 3:   The release of a Call/Connection from A-Party for a redirected call where the call segments are in the "Stable_2_Party" state and in the "Forward" states is given in figure 29.
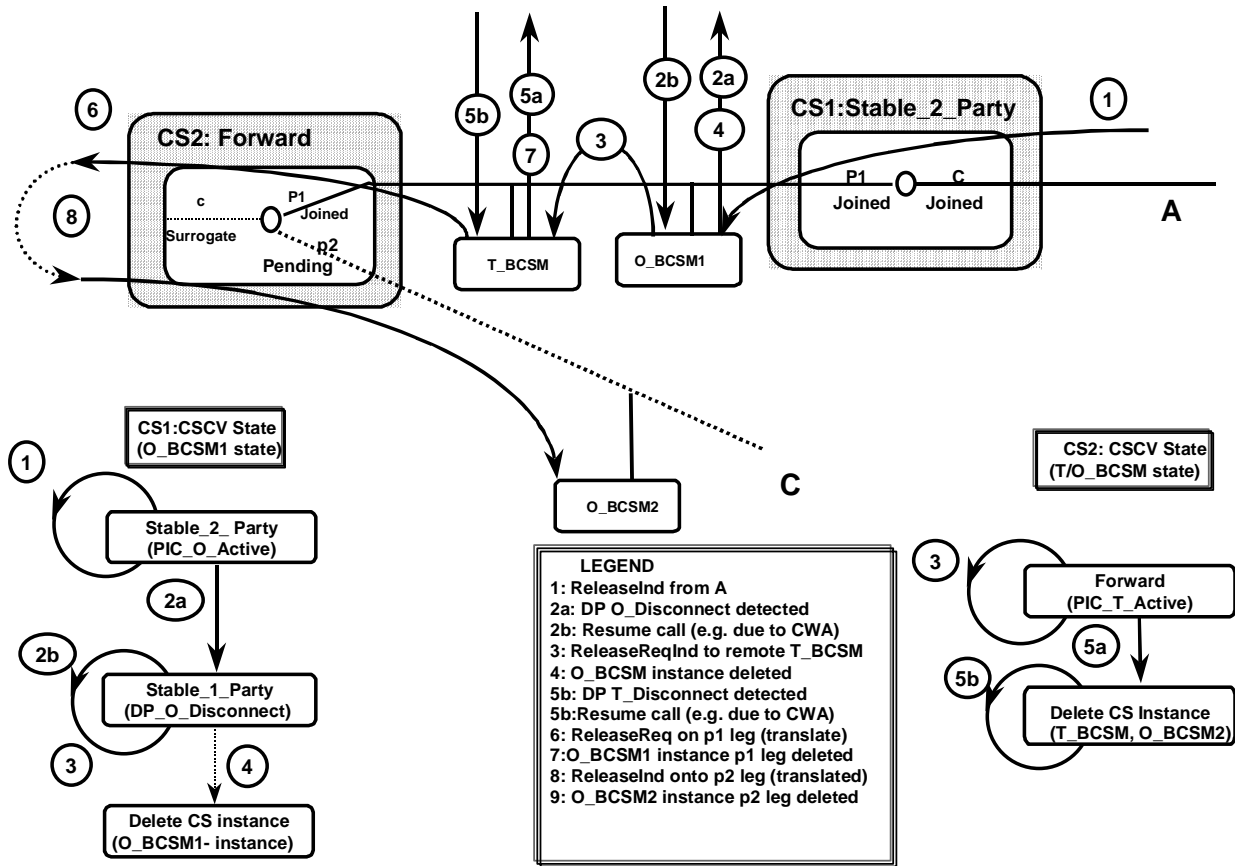
Figure 29: Example 3: Release of Call/Connection from A-Party where the call segments are in the "Stable_2_Party" state and in the "Forward" state

EXAMPLE 4: The release of a Call/Connection for a redirected call where the call segments are in the "Stable_2_Party" state and in the "Stable_Multi_Passive_Party" states is given in figures 30 and 31.
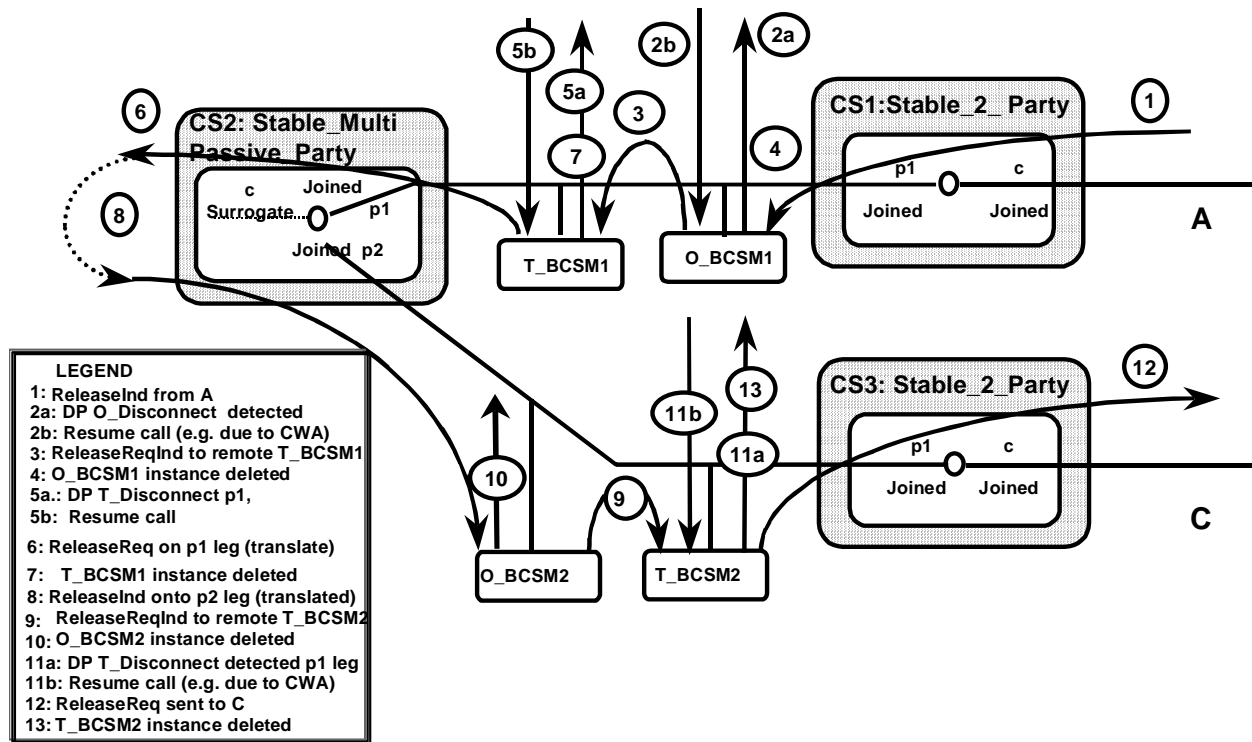
Figure 30: Example 4: Release of Call/Connection from A-party where the call segments are in the "Stable_2_Party" state and in the "Stable_Multi_Passive_Party" state (information flow)

EXAMPLE 4 (continued): The CSCV state transitions for release of a Call/Connection for a redirected call where the call segments are in the "Stable_2_Party" state and in the "Stable_Multi_Passive_Party" states are given in figure 31.
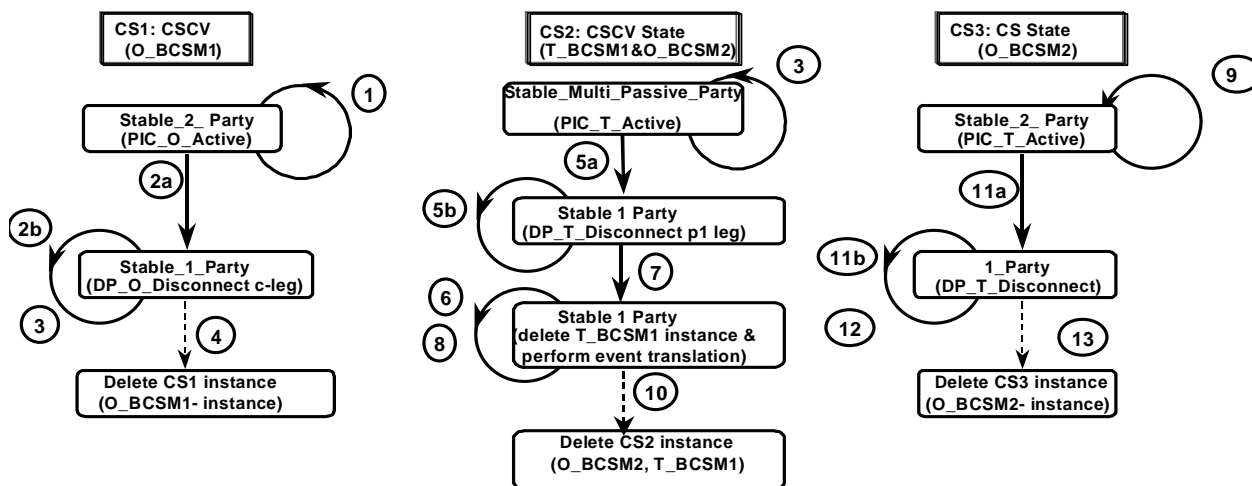
Figure 31: Example 4 (Cont'd): Release of Call/Connection from A-party where the call segments are in the "Stable_2_Party" state and in the "Stable_Multi_Passive_Party" state (information flow)

6.6.3.4.2.5   DisconnectLeg Operation

A DisconnectLeg operation for a controlling and passive leg shall physically release the specified leg from the connection point towards the remote user.

The following general DisconnectLeg and BCSM Transition Rules applies:

a) When a DisconnectLeg(p) is performed on the last joined passive leg in a CS for which an association with the joined controlling leg exists, the BCSM is suspended according to the rule b).

b) If call processing is already suspended at a DP reported for the c-leg, it remains suspended at that DP; otherwise a transition to O_/T_Mid_Call DP for leg c is performed (due to the applied CPH operation).

c) When a disconnectLeg (c) is performed on the joined controlling leg in a CS then the BCSM instance(s) attached to the passive leg(s) will be suspended at MidCall DP if not already suspended at a DP on the passive leg.

d) On receipt of a valid DisconnectLeg the leg is deleted immediately. When the last joined leg in the CS is released by SCF due to DisconnectLeg, the CS (including its associated BCSM instances) is deleted immediately (i.e. no call process suspension in this case).

An example depicting the release sequences as a result of a DisconnectLeg operation for the controlling (c) and passive legs of a Call/Connection for a normal call scenario where both call segments are in the "Stable_2_Party" state (with the associated O_BCSM and T_BCSM in the active state) is given in figure 32.

The information flows for the receipt of the following operations are illustrated:

-   The following actions occur when a DisconnectLeg (c) operation (for the controlling leg) is received in CS1 CSCV state "Stable_2-Party" from the SLP in the SCF:

    -   a ReleaseReq signal is sent toward the A-side with a cause value "disconnect controlling leg"; and

    -   the O_BCSM goes to the appropriate O_MidCall DP according to rule c) above.

-   The following actions occur when subsequent a DisconnectLeg (p1) operation (for the passive leg) is received in CS1 CSCV state "Stable_1_Party" from the SLP in the SCF:

    -   a ReleaseReqInd signal is sent to remote T_BCSM with a cause value "disconnect passive leg"; and

    -   the O_BCSM state transition occurs according to rule d) above as the last joined leg in the CS is deleted.

On receipt of a
DisconnectLeg(c) an
ReleaseReq with cause
"disconnect controlling
leg" is sent to the A-side

The BSCM shall transit into
the appropriate
DP_MidCall

LEGEND
1:   DisconnectLeg c
2a :ReleaseReq to A
2b: Resume call (e.g. CWA)
3   DisconnectLeg p1
4: ReleaseReqInd to T_BCSM
5: O_BCSM instance deleted
6a: DP T_Disconnect p1 leg
6b: Resume call (e.g. CWA)
7:   ReleaseReq to B
8:   T_BCSM instance deleted

Figure 32: Example of Disconnect Leg of Call/Connection where call segments are both in "Stable_2_Party"

6.6.3.4.2.6   User interactions during the SSF-FSM "Monitoring" state

During the "Monitoring" state of the SSF-FSM it shall be possible to perform user interactions in order to send tones, announcements and display information. For details refer to clause 8.

6.6.3.4.2.7   Call Segment and associated BCSM states for CPH operations

The following principles apply for the Call Segment and associated BCSM states for CPH operations:

1)   If a CPH operation (SplitLeg, DisconnectLeg, MergeCallSegments, MoveCallSegment or MoveLeg) is received in the Monitoring state, the FSMs for the involved Call Segments shall first go to the "Waiting for Instruction" state while the associated BCSM instances within the involved Call Segments shall move from the O/T PIC to the corresponding O/T_Mid_Call DP in order to handle subsequent EDP rearming. It shall be noted that when the BCSM instance is in a DP it shall stay in this DP where appropriate after processing of the CPH operation. This involves that only some PIC transitions to MidCall DP's are possible as described in the templates of the operations.

2)   The receipt of a CPH operation received and/or processed in the state Waiting for Instructions shall not cause the change of the state Waiting for Instructions.

3)   For these CPH operations which create for the controlling leg a new BCSM, only this BCSM shall be put into the DP-O/T_Mid_Call, the BCSM states of the other involved CS should transit from the PIC into the corresponding DP state or if already suspended remain in the DP wait state, the associated CS SSF_FSM for the newly created BCSM and the other involved CS shall be put into the "Waiting for Instructions" state so that the EDPs can be rearmed.

4)   All CPH operation sequences shall be finalized by an operation which changes the state into Monitoring (e.g. ContinueWithArgument, Connect).

6.6.3.4.2.8   "Forward" and "Stable_Multi_Passive_Party" connection view behaviour principles

The connection view for Forward and Stable_Multi_Passive_Party state behaves differently for the case where an operation is received from Terminating Setup (e.g. Call forward service) or Stable_1_Party (e.g. meet me conference service). Depending on the situation the signalling events received from one party (one leg) may have to be relayed to the other party.

The "Stable_Multi_Passive_Party" CSCV state can be entered via:

a) Originating_1_Party_Setup -> Stable_1_Party -> Forward -> Stable_Multi_Passive_Party, e.g. SCF initiated call:
    A SCF initiated call (due to an InitiateCallAttempt and Connect operation).
    In this CSCV state the passive legs shall both have O_BCSMs connected to them. The receipt of a SetupRespConf, CallProgressReqInd or DataReqInd from the connected-to leg is **not relayed** to the initial created leg ("outgoing" leg).
    As in that case there is only two passive legs in the CS, the ReleaseReqInd will be transmitted to the initial party.

b) Terminating_Setup -> Forward -> Stable_Multi_Passive_Party, e.g. User initiated forwarded call:
    A call diversion at the terminating party (instantiated by a T_BCSM).
    In this CSCV state the passive leg connected to the initial party is associated with a T_BCSM, while the passive leg connected to the diverted-to party is associated with an O_BCSM. The receipt of a SetupRespConf, CallProgressReqInd or DataReqInd from the diverted to leg **is relayed** to the initial created leg ("incoming" leg). The ReleaseReqInd will be transmitted to the initial party.

NOTE: From the point of signalling interaction is only the view of one SLP considered, which is assumed to have the full visibility of the call and any signalling interworking aspects in case of multiple SLPIs with different visibility of the call is not handled.

6.6.3.4.2.9   "Stable_2_Party" and "Stable_1_Party" connection view behaviour principles

Since the Stable_1_Party CSCV state can be entered from the "Stable_2-Party" state as a result of the DisconnectLeg leg c operation, the receipt of a SetupConf, or a DataInd (from B-party) in the T_BCSM may not cause the sending of a corresponding signal (a SetupRespConf, or a DataReqInd) to the remote half call O_BCSM. A SetupConf, or DataInd can be received, for example due to glare (DisconnectLeg c for release of B-Party is received in CCF/SSF from SCF at the same time as SetupConf/DataInd is received from B-party).

However in the Stable_1_Party CSCV state when an O_BCSM is connected to the passive leg, the receipt of a SetupRespConf, or a ServiceFeatureIndicationReqInd (from B-party) in the O_BCSM causes the sending of a corresponding DP if armed.

6.6.3.4.3   Call Segment Connection View (CSCV) State Description

In case of any discrepancy between each of the CSCV state transition descriptions below and the Operation Procedures descriptions in clause 11, the Operation Procedures descriptions shall take precedence.

6.6.3.4.3.1   "Null" Call Segment Connection View State

The "Null" CSCV represents a condition where call processing is not active.
There is no controlling leg or passive leg connected to the connection point.

The characteristics of the "Null" CSCV state follow:

a) *Relationship with the BCSM*:

The "Null" CSCV state is associated with the call processing in O_Null/T_Null.

> **CSCV state "Null" encompasses the following BCSM PICs and DPs:**
>
> **O_BCSM**: DPs: none PICs: O_Null
> **T_BCSM:** DPs: none PICs: T_Null

b) *Entry Events*:

Processing of a valid CreateCallSegments operation or InitiateCallAttempt operation received from SCF, or call processing in O_Null/T_Null, i.e. before IN triggering occurs (before sending of an InitialDP operation to SCF).

c) *Exit Events:*

Summary of CSCV state-related SSF events: Origination_Attempt.
i) To the "Originating_1_Party_ Setup" CSCV state. Processing of a valid InitiateCallAttempt operation received from SCF. ii) To the "Originating_Setup" CSCV state. Processing of a valid SetupInd signal received in O_BCSM from Signalling Control. iii) To the "Terminating_Setup" CSCV state. - Processing of a valid SetupReqInd signal received in T_BCSM from remote O_BCSM.

6.6.3.4.3.2   "Originating_Setup" Call Segment Connection View State

The "Originating_Setup" CSCV state represents e.g. an originating two-party call in the setup phase.
The call segment instance has a joined controlling leg and a pending passive leg with an associated O_BCSM.

The characteristics of the "Originating_Setup" CSCV state follow:

a)   Relationship with the BCSM:

controlling *leg* = c:
    legStatus = joined

passive *leg* = p1:
    legStatus = pending

CSCV state "Origination_Setup" encompasses the following BCSM PICs and DPs:
O_BCSM: DPs: Origination_Attempt TDP for leg c, Origination_Attempt_Authorized DP for leg c, Origination_Attempt_Denied DP for leg c, Collected_Information DP for leg c, Analysed_Information DP for leg c, O_Mid_Call (Send_Call) for leg c, Route_Select_Failure DP for leg p, Authorize_Route_Failure DP for leg p, O_Called_Party_Busy DP for leg p, O_No_Answer DP for leg p, O_Abandon DP for leg c or O_Disconnect DP for leg p. PICs: Authorize_Origination_Attempt, Collect_Information, Analyse_Information, Select_Route, Authorize_Call_Setup, Send_Call.

b) *Entry Events***:**

- Detection of the Origination_Attempt event as described in O_Null.

- See the exit events for the "Stable_2_Party" CSCV state.

- See the exit events for the "1_Party" CSCV state.

c) *Exit Events***:**

Summary of CSCV-related SSF events:
O_BCSM: Origination_Attempt TDP for leg c, Origination_Attempt_Authorized DP for leg c, Origination_Attempt_Denied DP for leg c, Collected_Information DP for leg c, Analysed_Information DP for leg c or leg p, O_Mid_Call DP (Send_Call) for leg c, O_Term_Seized DP for leg p, Route_Select_Failure DP for leg p, Authorize_Route_Failure DP for leg p, O_Called_Party_Busy DP for leg p, O_No_Answer DP for leg p, O_Abandon DP for leg c or O_Disconnect DP for leg p.

Direct Transitions from the "Originating_Setup" CSCV state:

i) To the "Stable_2_Party" CSCV state

- Detection of the O-Term_Seized DP for leg p in the O_BCSM (as described in PIC O_Alerting) is received from the remote T_BCSM). If no such DP is encountered the transition is detected at the O-Answer DP for leg p (as described in the PIC O_Active for the direct answer case).

- Processing of a ImportLeg for leg p due to a valid MoveLeg for leg p or MergeCallSegments (as target CS) operation received from SCF, in response to any DP that is allowed to be reported in the "Originating_Setup" CSCV state.

- Processing of a ImportLeg for leg p due to a valid MoveLeg for leg p or MergeCallSegments (as target CS) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

ii) To the deletion of the "Origination_Setup" CSCV state (deletion of CS instance)

- Processing of a valid ReleaseCall operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a valid ReleaseCall operation received from SCF, in response to any DP that is allowed to be reported in the "Originating_Setup" CSCV state.

- Processing of a ExportLeg(c) due to a valid MoveLeg(c) or MergeCallSegments (as Source CS) operation received from SCF, in response to a DP that is allowed to be reported in the "Originating_Setup" CSCV state.

- Processing of a ExportLeg(c) due to a valid MoveLeg(c) or MergeCallSegments (as Source CS) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA). Operation is allowed when call processing is suspended at a DP before the O_BCSM has sent a SetupReqInd (see note 2).

- Processing of a valid Continue or ContinueWithArgument operation received from SCF, in response to an Origination_Attempt_Denied DP for leg c, Authorize_Route_Failure DP for leg p or Route_Select_Failure DP for leg p, or O_Called_Party_Busy DP for leg p, or O_No_Answer DP for leg p, or O_Disconnect DP for leg p or O_Abandon DP for leg c (see note 1).

- Processing of a valid DisconnectLeg operation for leg c received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA) (see note 2).

- Processing of a valid DisconnectLeg(c) operation for leg c received from SCF, in response to any DP that is allowed to be reported in the "Originating_Setup" CSCV state (see note 2).

NOTE 1: When a release event is detected on the last joined c-leg in the CS, the leg state is changed (from "joined" to "surrogate") without changing the CSCV state and the CS is deleted upon resumption of call processing.

NOTE 2: When DisconnectLeg or Moveleg or MergeCallSegments is performed on the last joined c-leg in the CS, the CS is deleted.

iii) To the "Origination_Setup" CSCV state

NOTE 3: Will remain in the CSCV state for any valid DP and PIC that comprises the definition of this CSCV state.

6.6.3.4.3.3   "Terminating_Setup" Call Segment Connection View State

The "Terminating_Setup" CSCV state represents e.g. a terminating two-party call in the setup phase.
The characteristics of the "Terminating_Setup" CSCV state follow: This state represents a call segment instance with a
pending controlling leg and a joined passive leg with an associated T_BCSM.

The characteristics of the "Terminating_Setup" CSCV state follow:

a) Relationship with the BCSM:

>   controlling leg = c:
>   legStatus = pending
>
>   passive leg = p1:
>   legStatus = joined

CSCV state "Terminating_Setup" encompasses the following BCSM PICs and DPs:

T_BCSM:
DPs: Terminating_Attempt TDP for leg p, Terminating_Attempt_Authorized DP for leg p, Termination_Attempt_Denied
DP for leg p, Facility_Selected_and_Availlable DP for leg c, T_Busy DP for leg c, T_No_Answer DP for leg c,
T_Abandon DP for leg p.

PICs: Authorize_Termination_Attempt, Select_Facility and Present_Call.

---

b) Entry Events:

- • Detection of the Termination_Attempt event as described in T_Null.

- • See the exit events for the "Stable_2_Party".

c) Exit Events:

Summary of CSCV state-related SSF events:

T_BCSM: Terminating_Attempt TDP for leg p, Terminating_Attempt_Authorized DP for leg p,
Termination_Attempt_Denied DP for leg p, Facility_Selected_and_Availlable DP for leg c, T_Busy DP for leg c,
T_No_Answer DP for leg c T_Abandon DP for leg p.

Direct Transitions from the "Terminating_Setup" CSCV state:

i) To the deletion of the "Terminating_Setup" CSCV state (deletion of CS instance)

- Processing of a valid ReleaseCall operation received from SCF, in response to a to any DP that is allowed to be reported in the "Terminating_Setup" CSCV state.

- Processing of a valid ReleaseCall operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a valid DisconnectLeg(p) operation for leg p received from SCF, in response to a DP that is reported in the "Terminating_Setup" CSCV state (see note 2).

- Processing of a valid DisconnectLeg(p) operation for leg p received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA) (see note 2).

- Processing of a valid Continue or ContinueWithArgument operation received from SCF in response to a Termination_Attempt_Denied DP for leg p, T_Abandon DP for leg p, T_Busy DP for leg c, T_No_Answer DP for leg c (see note 1).

NOTE 1: When a release event is detected on the last joined p-leg in the CS, the leg state is changed (from "joined" to "pending") without changing the CSCV state and the CS is deleted upon resumption of call processing.

NOTE 2: When DisconnectLeg is performed on the last joined p-leg in the CS, the CS is deleted.

ii) To the "Stable_2_Party" CSCV state

- Detection of the Call_Accepted DP (as described in PIC T_Alerting) or the T_Answer DP (as described on PIC T_Active) as applicable in the direct answer case.

- Processing of a ImportLeg for leg c due to a valid MoveLeg for leg c or MergeCallSegments (as target CS) operation received from SCF, in response to Terminating_Attempt TDP for leg p, Terminating_Attempt_Authorized DP for leg p, Termination_Attempt_Denied DP for leg p, Facility_Selected_and_Availlable DP for leg c, T_Busy DP for leg c, T_No_Answer DP for leg c.

NOTE 3: The existing pending c-leg in Terminating_Setup CSCV is removed when another c-leg is imported.

iii) To the "Forward" CSCV state

- Processing of a valid Connect operation received from SCF in response to a DP that is reported in the "Terminating_Setup" CSCV state. Processing of a valid SelectFacility operation with DestinationNumberRoutingAddress parameter received from SCF in response to any DP that is allowed to be reported in the "Terminating_Setup" CSCV state.

iv) To the "Stable_Multi_Passive_Party" CSCV state

- Processing of a ImportLeg for leg p due to a valid MoveLeg for lep c or MergeCallSegments (as target CS) operation received from SCF, in response to Terminating_Attempt TDP for leg p, Terminating_Attempt_Authorized DP for leg p, Termination_Attempt_Denied DP for leg p, Facility_Selected_and_Availlable DP for leg c, T_Busy DP for leg c, T_No_Answer DP for leg c.

NOTE 4: The existing pending c-leg in Terminating_Setup CSCV is removed when another p-leg is imported.

v) To the "Terminating_Setup" CSCV state

- Processing of a valid SelectFacility operation without DestinationNumberRoutingAddress parameter received from SCF in response to a DP that is reported in the "Terminating_Setup" CSCV state.

NOTE 5: Will remain in the CSCV state for any valid DP and PIC that comprises the definition of this CSCV state.

6.6.3.4.3.4   "Stable_2_Party" Call Segment Connection View State

The "Stable_2_Party" CSCV state represents a stable or clearing two-party call, and is either an originating or a terminating call from the perspective of the controlling user with a joined controlling leg and a "joined" passive leg with either an O_BCSM or a T_BCSM associated with it.

The characteristics of the "Stable_2_Party" CSCV state follow:

a) Relationship with the BCSM:

   controlling leg = c:
    legStatus = joined;

   passive leg = p1:
    legStatus = joined.
   CSCV state "Stable_2_Party" encompasses the following BCSM PICs and DPs:

```
O_BCSM:
DPs leg p1: O_Term_Seized DP, O_Answer DP, O_Mid_Call DP {O_Active}, O_Mid_Call DP {O_Suspended},
O_Suspend DP, O_Re-answer DP.
DPs leg c: O_Mid_Call DP {O_Alerting }, O_Mid_Call DP {O_Active}, O_Mid_Call DP {O_Suspended},
PICs: O_Alerting, O_Active, O_Suspended,

T_BCSM:
DPs leg c: Call_Accepted DP, T_Answer DP, T_Suspend DP, T_Re_Answer DP, T_Mid_Call DP {T_Active}.
DPs leg p1: T_Mid_Call DP {T_Active}.
PICs: T_Alerting, T_Active, T_Suspended.
```

b) *Entry Events:*

See the exit events for CSCV states:

- "Originating_ Setup",

- "Terminating_ Setup";

- "1_Party";

- "Stable_Multi_Party";

- "Stable_1_Party".

c) *Exit Events:*

Summary of CSCV state-related SSF events:

O_BCSM:
Switch_Hook_Flash_Immediate trigger, BRI_Feature_Activation_Indicator trigger, second dial tone associated with the Switch_Hook_Flash_Specified_Code trigger, O_Mid_Call DP{O_Alerting } for leg c, O_Mid_Call DP (O_Active,O_Suspended} for leg c and leg p, O_Mid_Call DP {T_Active} for leg c and leg p, O_Abandon DP for leg c, O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p,O_Disconnect DP for leg c and leg p,

T_BCSM:
Switch_Hook_Flash_Immediate trigger, BRI_Feature_Activation_Indicator trigger, second dial tone associated with the Switch_Hook_Flash_Specified_Code trigger T_Mid_Call DP {T_Active} for leg c and leg p, Call_Accepted DP for leg c, T_Answer DP for leg c, T_Suspend DP for leg c, T_Re_Answer DP for leg c, T_Abandon for leg p and T_Disconnect DP for leg c and leg p.

Direct Transitions from the "Stable_2_Party" CSCV state:

i) To the deletion of the "Stable_2_Party" CSCV state (deletion of CS instance)

- Processing of a valid ReleaseCall operation received from SCF, in response to:

  ▪ an O_Mid_Call DP or T_Mid_Call DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or O_Mid_Call DP{O_Alerting} for leg c, O_Mid_Call DP {O_Active,O_Suspended} for leg c and leg p, T_Mid_Call DP {T_Active} for leg c and leg p;

  ▪ O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p; and
  Call_Accepted DP for leg c, T_Answer DP for leg c, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a valid ReleaseCall operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a valid Continue or ContinueWithArgument operation received from SCF, in response to detection of the O_Disconnect DP and O_Abandon DP for leg c sent from the O_BCSM or detection of the T_Disconnect DP for leg c sent from the T_BCSM (see note 1).

NOTE 1: When a release event is detected on the last joined c-leg in the CS, the leg state is changed (from "joined" to "surrogate") without changing the CSCV state and the CS is deleted upon resumption of call processing.

ii) To the "Originating Setup" CSCV state

- Detection of the O_Disconnect DP for leg p sent from the O_BCSM on receipt of a ReleaseReqInd signalling event from the remote T_BCSM (see note 2).

- Detection of the O_No_Answer DP for leg p sent from the O_BCSM on receipt of a ReleaseReqInd signalling event from the remote T_BCSM or time-out of the no reply timer.

- Detection of the O_Called_Party_Busy DP for leg p sent from the O_BCSM on receipt of a ReleaseReqInd signalling event from the remote T_BCSM.

- Detection of a Route_Select_Failure DP due to a Release event which generates a "Route_Failure" event causing a transition from PIC Alerting to PIC Select_Route.

NOTE 2: Distinction between call setup and call clearing is possible, i.e. between an implicit (not modelled) "2-party pending" CSCV state and an "originating setup" CSCV state (both consisting of a joined controlling leg and a pending passive leg).

The state "2-party pending" is reached from the Stable 2-party state with an O-BCSM when a release event is received on the passive leg after the O-Answer DP has been encountered, i.e. after the Setup (call setup) has been confirmed. The "originating setup" CSCV state applies before answer.
The DisconnectLeg(p) operation is accepted in this implicit "2 party-pending" CSCV state but not in the "originating setup" CSCV state.

iii) To the "Terminating Setup" CSCV state

- Detection of the T_No_Answer DP for leg c sent from the T_BCSM (on receipt of a ReleaseInd signalling event or due to time-out of the no reply timer).

- Detection of the T_Busy DP for leg c sent from the T_BCSM.

iv) To the "1_Party" CSCV state

- Processing of a valid DisconnectLeg (p) operation for *leg* p received from SCF, in response to:

  - an O_Mid_Call or T_Mid_Call DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or O_Mid_Call DP O_{Alerting} for leg c, or O_Mid_Call DP {O_Active,O_Suspended} for leg c and leg p, or T_Mid_Call DP {T_Active} for leg c and leg p).

  - O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, Call_Accepted DP for leg c, T_Answer DP for leg c, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a valid DisconnectLeg(p) operation for leg p received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a ExportLeg(p) due to a valid MoveLeg(p) or SplitLeg(p) operation received from SCF, in response to:

  - an O_Mid_Call or T_Mid_Call DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger,
    O_BCSM: O_Mid_Call DP{O_Alerting} for leg c, O_Mid_Call DP { O_Active,O_Suspended} for leg c and leg p, or
    T_BCSM: T_Mid_Call DP {T_Active} for leg c and leg p).

  - O_BCSM: O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p,
    T_BCSM: Call_Accepted DP for leg c, T_Answer DP for leg c, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a ExportLeg(p) due to a valid MoveLeg(p) or SplitLeg(p) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Detection of the T_Disconnect DP for leg p sent from the T_BCSM.

- Detection of the T_Abandon DP for leg p sent from the T_BCSM.

v) To the "Stable_2_Party" CSCV state

NOTE 3: Will remain in the CSCV state for any valid DP and PIC that comprises the definition of this CSCV state.

vi) To the "Stable_Multi_Party" CSCV state

- Processing of a ImportLeg(p) due to a valid MoveLeg(p) or MergeCallSegments (as Target CS) operation received from SCF, in response to:

  ▪ an OMidCall or TMidCall DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger,
    O_BCSM: O_Mid_Call DP{O_Alerting} for leg c, O_Mid_Call DP{ O_Active,O_Suspended} for leg c and leg p, or
    T_BCSM: T_Mid_Call DP {T_Active} for leg c and leg p).

  ▪ O_BCSM: O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p,
    T_BCSM: Call_Accepted DP for leg c, T_Answer DP for leg c, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a ImportLeg(p) due to a valid MoveLeg(p) or MergeCallSegments(as Target CS) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

vii)To the "Stable_1_Party" CSCV State

- Processing of a valid DisconnectLeg operation for *leg* c received from SCF, in response to:

  ▪ an O_Mid_Call or T_Mid_Call DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger,
    O_BCSM: O_Mid_Call DP{O_Alerting} for leg c, O_Mid_Call DP{ O_Active,O_Suspended} for leg c and leg p, or
    T_BCSM: T_Mid_Call DP {T_Active} for leg c and leg p.

  ▪ O_BCSM: O_Term_Seized DP for leg p, O_Answer DP for leg p,, O_Suspend DP for leg p, O_Re-answer DP for leg p,
    T_BCSM: Call_Accepted DP for leg c, T_Answer DP for leg c, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a valid DisconnectLeg operation for leg c received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a ExportLeg(c) due to a valid MoveLeg(c) or SplitLeg(c) operation received from SCF, in response to:

  ▪ an O_Mid_Call or T_Mid_Call DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or O_Mid_Call DP{O_Alerting} for leg c, O_Mid_Call DP{O_Active, O_Suspended} for leg c and leg p or T_Mid_Call DP {T_Active} for leg c and leg p.

  ▪ O_BCSM: O_Answer DP for leg p, O_Term_Seized DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p,
    T_BCSM: Call_Accepted DP for leg c, T_Answer DP for leg c, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a ExportLeg(c) due to a valid MoveLeg(c) or SplitLeg(c) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

6.6.3.4.3.5   "1_Party" Call Segment Connection View State

The "1_Party" CSCV state represents a 1-party call with a joined controlling leg status and no passive legs. The O_BCSM or T_BCSM is associated with the controlling leg since no passive leg is connected to the CP.

The characteristics of the "1_Party" CSCV state follow:

a) Relationship with the BCSM:

> controlling *leg* = c
>   *legStatus* = joined

CSCV state "1_Party" encompasses the following BCSM PICs and DPs:

O_BCSM:
DPs leg c: O_Mid_Call DP{ O_Alerting }, O_Mid_Call DP {O_Active}, O_Abandon DP, O_Disconnect DP,
PICs: O_Alerting, O_Active.

T_BCSM:
DPs leg c: Call_Accepted DP, T_Answer DP, T_Busy DP, T_No_Answer DP, T_Suspend DP, T_Re_Answer DP,
T_Mid_Call DP {T_Active}, T_Disconnect DP(see note 3)
DPs leg p: T_Abandon DP, T_Disconnect DP (see note 2)
PICs: T_Alerting, T_Active, T_Suspended

NOTE 1: Leg c represents a calling party ("incoming" leg) if O_BCSM and a called party ("outgoing" leg) if T_BCSM.

NOTE 2: Transition from "Stable-2_Party" to "1_Party" at release from originating call half (calling party/entity disconnect).

NOTE 3: CS is deleted upon resumption of call processing.

b) Entry Events:

See the exit events for CSCV states:

- Stable_2_Party.

- Stable_Multi_Party.

c) Exit Events:

Summary of CSCV state-related SSF events:

O_BCSM: leg c: O_Mid_Call DP{ O_Alerting }, O_Mid_Call DP {O_Active}, O_Abandon DP, O_Disconnect DP,
PICs: O_Alerting, O_Active.

T_BCSM: leg c, Call_Accepted DP, T_Answer DP, T_Busy DP, T_No_Answer DP, T_Suspend DP, T_Re_Answer DP,
T_Mid_Call DP {T_Active}, T_Disconnect DP.
leg p: T_Abandon DP, T_Disconnect DP.

Direct Transitions from the "1_Party" CSCV state:

i) To the deletion of the "1_Party" CSCV state (deletion of CS instance see note 4)

- Detection of the O_Abandon DP for leg c (see note 4).

- Detection of the O_Disconnect DP for leg c sent from the O_BCSM (see note 4).

- Detection of the T_Busy DP for leg c sent from the T_BCSM (see note 4).

- Detection of the T_No_Answer DP for leg c sent from the T_BCSM or on the expiry of a no reply timer event (see note 4).

- Detection of the T_Disconnect DP for leg c sent from the T_BCSM (see note 4).

- Processing of a valid ReleaseCall operation received from SCF.

- Processing of a valid ReleaseCall operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a valid DisconnectLeg operation for leg c received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from a CS of the same CSA) (see note 5).

- Processing of a ExportLeg(c) due to a valid MoveLeg(c) or MergeCallSegments (source CS) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from a CS of the same CSA).

- Processing of a valid Continue or ContinueWithArgument operation received from SCF, in response to an T-Abandon DP for leg p, T_Disconnect DP for leg p (see notes 4 and 6).

- Processing of a valid Continue or ContinueWithArgument operation received from, in response to a T_Busy DP for leg c T_No_Answer DP for leg c, T_Disconnect DP for leg c (see note 4).

NOTE 4: When a release event is detected on the last joined c-leg in the CS, the leg state is changed (from "joined" to "surrogate") without changing the CSCV state and the CS is deleted upon resumption of call processing.

NOTE 5: When DisconnectLeg is performed on the last joined c-leg in the CS, the CS is deleted.

NOTE 6: This is a special call clearing case applicable after a transition from "Stable-2_Party (T_BCSM)"_ to "1_Party" due to a release from originating call half.

ii) To the "1_Party" CSCV state

- Detection of the Call_Accepted DP, T_Answer DP for leg c sent from the T_BCSM.

- Detection of the T_Mid_Call DP{T_Active, T_Suspended} for leg c sent from the T_BCSM.

- Detection of the O_Mid_Call DP {O_Active} for leg c sent from the O_.

NOTE 7: Will remain in the CSCV state for any valid DP and PIC that comprises the definition of this CSCV state.

iii) To the "Stable_2_Party" CSCV state

- Processing of a ImportLeg for leg p due to a valid MoveLeg for leg p or MergeCallSegments (as target CS) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from a CS of the same CSA).

iv) To the "Originating_Setup" CSCV state

- Processing of a valid Connect operation received from the SCF, in response to O_Mid_Call DP{ O_Alerting }, O_Mid_Call DP {O_Active} for leg c.

- Processing of a valid Connect operation received from the SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, a SCF time supervision, an user to service information or any other valid DP or notification/event received from a CS of the same CSA).

6.6.3.4.3.6   "Originating_1_Party_Setup" Call Segment Connection View State

The "Originating_1_Party_Setup" CSCV state represents a 1-party call with a surrogate controlling leg and a pending passive leg with which an O_BCSM is associated being originated on behalf of the network (i.e. the controlling leg has a *legStatus* = surrogate).

The characteristics of the "Originating_1_Party_Setup" CSCV state follow:

a) Relationship with the BCSM:

> controlling *leg*: = c
>   *legStatus* = surrogate
> passive *leg* = p1:
>   *legStatus* = pending

CSCV state "Originating_1_Party_Setup" encompasses the following BCSM PICs and DPs:

O_BCSM:
DPs: leg p1
Origination_Attempt TDP, Origination_Attempt_Authorized DP, Origination_Attempt_Denied DP, Collected_Information DP, Analysed_Information DP, *O_Mid_Call (Send_Call),*
Route_Select_Failure DP, Authorize_Route_Failure DP, O_Called_Party_Busy DP, O_No_Answer DP, O_Disconnect DP.
PICs: *Authorize_Origination_Attempt, Collect_Information,* Analyse_Information, Select_Route, Authorize_Call_Setup, Send_Call.

NOTE: Regarding the DPs and PICs as shown for p1 leg in italic, it is outside the scope of this capability set to define any application related to this CSCV state.

---

b) Entry Events:

See the exit events for the CSCV states:

- "Null".

- "Stable_1_Party".

c) Exit Event

Summary of CSCV state-related SSF events:

O_BCSM: leg p1: Analysed_Information DP, Route_Select_Failure DP, Authorize_Route_Select_Failure DP, O_Called_Party_Busy DP, O_No_Answer DP, O_Disconnect DP.

Direct Transitions from "Originating_1_Party_Setup" CSCV state:

i)   To the deletion of the "Originating_1_Party_Setup" CSCV state (deletion of CS instance)

-   Processing of a valid ReleaseCall operation received from SCF, in response to any DP that is allowed to be reported in the Originating_1_Party_Setup CSCV state.

-   Processing of a valid ReleaseCall operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

-   Processing of a valid Continue or ContinueWithArgument operation received from SCF, in response to an Authorize_Route_Failure DP for leg p, Route_Select_Failure DP for leg p, O_Called_Party_Busy DP for leg p, O_No_Answer DP for leg p or O-Disconnect DP for leg p.

ii) To the "Stable 1-Party" CSCV state

-   Detection of the O_Term_Seized DP for leg p on the O_BCSM
    If no such message is received the transition is detected at the O-Answer DP for leg p as answer is received(direct answer case).

iii) To the "Originating_1_Party_Setup" CSCV state

NOTE: Will remain in the CSCV state for any valid DP and PIC that comprises the definition of this CSCV state.

6.6.3.4.3.7   "Stable_1_Party" Call Segment Connection View State

The "Stable_1_Party" CSCV state represents a 1-party call with a surrogate controlling leg and a joined passive leg with either an O_BCSM or T_BCSM associated with it, that is in a stable clearing phase.

The characteristics of the "Stable_1_Party" CSCV state follow:

a)   Relationship with the BCSM

> controlling *leg*:
>   *legStatus* = surrogate
> passive *leg* = p1:
>   *legStatus* = joined

CSCV state "Stable 1_Party" encompasses the following BCSM PICs and DPs:

O_BCSM,leg p1:O_Term_Seized DP, O_Answer DP, O_Suspend DP, O_Re-answer DP, O_Mid_Call DP {O_Alerting}
O_Mid_Call DP {O_Active}, O_Mid_Call DP (O_Suspended}.

PICs: O_Alerting, O_Active, O_Suspended.

T_BCSM: T_Mid_Call DP {T_Active} for leg p, T_Abandon DP for leg c, T_Disconnect for leg p.
PICs: T_Alerting, T_Active, T_Suspended.

---

b)   Entry Events:

See the exit events for CSCV states:

- "Stable_Multi_Passive_Party".

- "Stable_2_Party".

- "Stable_Multi_Party".

c)   Exit Events:

Summary of CSCV state-related SSF events:

O_Term_Seized DP for leg p, O_Mid_Call DP{, O_Active,O_Suspended} for leg p, T_Mid_Call DP {T_Active} for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, O_Disconnect DP and leg p, T_Disconnect DP for leg p, T_Abandon DP for leg p.

Direct Transitions from the "Stable 1-Party" CSCV state:

i) To the deletion of the "Stable_1_Party" CSCV state (deletion of CS instance)

- Detection of the T_Disconnect DP for leg p (see note 1).

- Detection of the T_Abandon DP for leg p sent from the T_BCSM (see note 1).

- Processing of a valid ReleaseCall operation received from SCF, in response to an O_Term_Seized DP for leg p, O_Mid_Call DP{, O_Alerting} for leg pO_Mid_Call DP {O_Active}for leg p, O_Mid_Call DP (O_Suspended} for leg p,, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, or T_Mid_Call DP {T_Active} for leg p, T-Abandon DP for leg p, T_Disconnect DP for leg p.

- Processing of a valid ReleaseCall operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a valid DisconnectLeg operation for *leg* p received from SCF, in response to an O_Term_Seized DP for leg p, O_Mid_Call DP{ O_Alerting} for leg p, O_Mid_Call DP {O_Active}for leg p, O_Mid_Call DP (O_Suspended} for leg p,, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p or
  T_Mid_Call DP {T_Active} for leg p, T-Abandon DP for leg p, T_Disconnect DP for leg p (see note 2).

- Processing of a valid DisconnectLeg(p) operation for leg p received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a ExportLeg(p) due to a valid MoveLeg(p) or MergeCallSegments (source CS) operation received from SCF, in response to an O_Term_Seized DP for leg p, O_Mid_Call DP{ O_Alerting,} for leg p, O_Mid_Call DP {O_Active}for leg p, O_Mid_Call DP (O_Suspended} for leg p, T_Mid_Call DP {T_Active} for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p or, T_Mid_Call DP {T_Active} for leg p.

- Processing of a ExportLeg(p) due to a valid MoveLeg(p) or MergeCallSegments (source CS) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

NOTE 1:   When a release event is detected on the last joined p-leg in the CS, the leg state is changed (from "joined" to "pending") without changing the CSCV state and the CS is deleted upon resumption of call processing.

NOTE 2:   When DisconnectLeg is performed on the last joined p-leg in the CS, the CS is deleted.

ii) To the "Forward" CSCV state

- Processing of a valid Connect operation received from SCF, in response to an O_Term_Seized DP for leg p, O_Mid_Call DP{O_Alerting} for leg p, O_Mid_Call DP {O_Active}for leg p, O_Mid_Call DP (O_Suspended} for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p or T_Mid_Call DP {T_Active} for leg p.

iii) To the "Stable_2_Party" CSCV state

- Processing of a ImportLeg for leg c due to a valid MoveLeg for leg c or MergeCallSegments (as target CS) operation received from SCF, in response to an O_Term_Seized DP for leg p, O_Mid_Call DP{ O_Alerting} for leg p, O_Mid_Call DP {O_Active}for leg p, O_Mid_Call DP (O_Suspended} for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p or
  T_Mid_Call DP {T_Active} for leg p.

- Processing of a ImportLeg for leg c due to a valid MoveLeg for leg c or MergeCallSegments (as target CS) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

iv) To the "Stable_Multi_Passive_party" CSCV state

- Processing of a ImportLeg for leg p due to a valid MoveLeg for leg p or MergeCallSegments (as target CS) operation received from SCF, in response to an O_Term_Seized DP for leg p, O_Mid_Call DP{ O_Alerting} for leg p, O_Mid_Call DP {O_Active}for leg p, O_Mid_Call DP (O_Suspended} for leg p, O_Answer for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p or
  T_Mid_Call DP {T_Active} for leg p.

- Processing of a ImportLeg for leg p due to a valid MoveLeg for leg p or MergeCallSegments (as target CS) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

v) To the "Originating_1_Party_Setup" CSCV state

- Detection of the O_Disconnect DP for leg p sent from the O_BCSM (on receipt of a ReleaseReqInd signalling event from the remote T_BCSM) (see note 1).

- Detection of the O_No_Answer DP for leg p sent from the O_BCSM (on receipt of a ReleaseReqInd signalling event from the remote T_BCSM or time-out of the no reply timer).

- Detection of the O_Called_Party_Busy DP for leg p sent from the O_BCSM (on receipt of a ReleaseReqInd signalling event from the remote T_BCSM).

- Detection of a Route_Select_Failure DP due to a Release event which generates a "Route_Failure" event causing a transition from PIC Alerting to PIC Select_Route.

NOTE 3: Distinction between call setup and call clearing is possible, i.e. between an implicit (not modelled) "clearing 1-party" CSCV state and an "originating 1 party setup" CSCV state (both consisting of a surrogate controlling leg and a pending passive leg).

The state "clearing 1-party" is reached from the Stable 1-party state with an O-BCSM when a release event is received on the passive leg after the O-Answer DP has been encountered, i.e. after the Setup (call setup) has been confirmed. The "originating 1-party setup" CSCV state applies before answer. This transition (release event) in "clearing 1-party" state leads to the termination of the CS, i.e. the CS is deleted upon resumption of call processing. This therefore obey the proposed rule that release/disconnection of the last joined leg in the CS causes the CS to be deleted. The DisconnectLeg(p) is accepted in this implicit "clearing 1- party" CSCV state, but not in the "originating 1-party setup" CSCV state.

vi) To the "Stable_1_Party" CSCV state

NOTE 4: Will remain in the CSCV state for any valid DP and PIC that comprises the definition of this CSCV.

6.6.3.4.3.8   "Forward" Call Segment Connection View State

The "Forward" CSCV state represents a forwarded call. Call processing for the first passive leg (*leg* p1) is in a stable whereas call processing for the second passive leg (*leg* p2) is in an originating call setup phase or clearing phase. This state has a surrogate controlling leg, a joined passive leg with which an O_BCSM or a T_BCSM is associated, and a pending passive leg with which an O_BCSM is associated. Note that the "surrogate" *legStatus* for the controlling leg indicates the charging relationship for the forwarded passive leg (*leg* p2) and that the "Forward" CSCV state is limited to two passive legs only. The characteristics of the "Forward" CSCV state follow.

The characteristics of the "Forward" CSCV state follow:

a) Relationship with the BCSM:

controlling *leg*:
legStatus = surrogate

passive *leg* = p1:
    legStatus = joined
passive *leg* = p2:
    legStatus = pending

CSCV state "Forward" encompasses the following BCSM PICs and DPs:

O_BCSM leg p1: O_Term_Seized DP for leg p1, O_Answer DP for leg p1, O_Suspend DP for leg p1, O_Re-answer DP for leg p1, O_Mid_Call DP {Alerting}, O_Mid_Call DP {Active} for leg p1, O_Mid_Call DP (O_Suspended} for leg p1.
PICs: O_Alerting, O_Active, O_Suspended,

T_BCSM leg p1: T-Abandon DP for leg p1, T_Mid_Call (Active) DP for leg p1, T-Disconnect DP for leg p1,
PICs: Present Call, T_Alerting, T_Active,

O_BCSM (leg p2 "forwarded leg":
      DPs: *Origination_Attempt TDP for leg p2, Origination_Attempt_Authorized DP for legp2,*
      Origination_Attempt_Denied DP for p2, Collected_Information DP for leg p2, Analysed_Information DP for leg
      p2, O_Mid_Call (Send_Call) for leg p2, Route_Select_Failure DP for leg p2, Authorize_Route_Failure DP for leg
      p2, O_Called_Party_Busy DP for leg p2, O_No_Answer DP for leg p2, or O_Disconnect DP for leg p2.
      PICs: *O_Null, Authorize_Origination_Attempt, Collect_Information,* Analyse_Information, Select_Route,
      Authorized_Call_Setup, Send_Call.

NOTE: Regarding the DPs and PICs as shown for the"forwarded leg" in italic, it is outside the scope of this capability set
        to define any application.

b) *Entry Events:*

See the exit events for the CSCV states:

- "Terminating Setup".

- "Stable_Multi_Passive_Party".

- "Stable 1-Party".

c) *Exit Events:*

Summary of CSCV state-related SSF events:

Origination_Attempt TDP, Origination_Attempt_Authorized DP for leg p2, Origination_Attempt_Denied DP for leg p2, Analysed_Information DP for leg p2, Route_Select_Failure DP for leg p2 or Authorize_Route_Failure DP for leg p2, O_Called_Party_Busy DP for leg p2, O_No_Answer DP for leg p2, O_Disconnect DP for leg p2.:
O_Term_Seized DP for leg p1, O_Answer DP for leg p1, O_Suspend DP for leg p1, O_Re-answer DP for leg p1, O_disconnect DP for leg p1, O_Mid_Call DP {Alerting}, O_Mid_Call DP {Active} for leg p1, O_Mid_Call DP (O_Suspended} for leg p1.
T-Abandon DP for leg p1, T_Mid_Call (Active) DP for leg p1, T_Disconnect DP for leg p1.

Direct Transitions from the "Forward" CSCV state:

i) **To the deletion of the "Forward" CSCV state (deletion of CS instance, see note 1)**

- Processing of a valid Continue or ContinueWithArgument operation received from SCF, in response to an Origination_Attempt_Denied DP for leg p2, Route_Select_Failure DP for leg p2 or Authorize_Route_Failure DP for leg p2, O_Called_Party_Busy DP for leg p2, O_No_Answer DP for leg p2, or O_Disconnect DP for leg p2.

- Processing of a valid Continue or ContinueWithArgument operation received from SCF, in response to a T_Disconnect DP for leg p1 if leg p1 associated with T_BCSM (see note 1).

- Processing of a valid Continue or ContinueWithArgument operation received from SCF, in response to an O_Disconnect DP, O_Called_Party_Busy DP and O_No_Answer DP for leg p1 if leg p1 associated with O_BCSM (see note 1).

- Processing of a valid ReleaseCall received from SCF, in response to an Origination_Attempt TDP, Origination_Attempt_Authorized DP for leg p2, Origination_Attempt_Denied DP for leg p2, Analysed_Information DP for leg p2,Route_Select_Failure DP for leg p2 or Authorize_Route_Failure DP for leg p2. O_Called_Party_Busy DP for leg p2, O_No_Answer DP for leg p2, or O_Disconnect DP for leg p2.

- Processing of a valid ReleaseCall operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification received from an associated CS of the same CSA).

- Processing of a valid DisconnectLeg operation for leg p1 the joined passive leg p1 is connected to a T_BCSM or O_BCSM) received from SCF, in response to an Origination_Attempt TDP, Origination_Attempt_Authorized DP for leg p2, Origination_Attempt_Denied DP for leg p2, Analysed_Information DP for leg p2, Route_Select_Failure DP for leg p2 or Authorize_Route_Failure DP for leg p2. O_Called_Party_Busy DP for leg p2, O_No_Answer DP for leg p2, or O_Disconnect DP for leg p2 (see note 2).

- Processing of a valid DisconnectLeg for leg p1 operation received from SCF (the joined passive leg p1 is connected to a T_BCSM or O_BCSM) which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA) (see note 2).

- Processing of a ExportLeg(p1) due to a valid MoveLeg(p1) operation received from SCF, in response to an Origination_Attempt TDP, Origination_Attempt_Authorized DP for leg p2, Origination_Attempt_Denied DP for leg p2, Analysed_Information DP for leg p2, Route_Select_Failure DP for leg p2 or Authorize_Route_Failure DP for leg p2, O_Called_Party_Busy DP for leg p2, O_No_Answer DP for leg p2, or O_Disconnect DP for leg p2 (see note 2).

- Processing of a ExportLeg(p1) due to a valid MoveLeg(p1) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information, or any other valid DP or notification/event received from an associated CS of the same CSA).

NOTE 1: When a release event is detected on the last joined p-leg in the CS, the leg state is changed (from "joined" to "pending") without changing the CSCV state and the CS is deleted upon resumption of call processing.

NOTE 2: When DisconnectLeg or MoveLeg is performed on the last joined p-leg in the CS, the CS is deleted.

ii) To the "Stable_Multi_Passive_Party" CSCV state

- Detection of the O_Term_Seized DP event sent from the O_BCSM for *leg* p2.
  If no such message is received the transition is detected at the O-Answer DP for leg p2 as answer is received(direct answer case).

iii) To the "Forward" CSCV state

- Processing of a valid Continue or ContinueWithArgument operation received from SCF, in response to an Origination_Attempt TDP, Origination_Attempt_Authorized DP for leg p2, Analysed_Information DP for leg p2.

- Processing of a valid Connect operation received from SCF, in response to Route_Select_Failure DP for leg p2 or Authorize_Route_Failure DP for leg p2, O_Called_Party_Busy DP for leg p2, O_No_Answer DP for leg p2, or O_Disconnect DP for leg p2.
  This operation overwrites any previous Connect operation information.

NOTE 3: Will remain in the CSCV state for any valid DP and PIC that comprises the definition of this CSCV.

6.6.3.4.3.9   "Stable_Multi_Passive_Party" Call Segment Connection View State

The "Stable_Multi_Passive_Party" CSCV state represents a transferred call. This state represents a transferred call with a surrogate controlling leg and two to N joined passive legs with each of which an O_BCSM or a T_BCSM is associated. The call between the involved passive legs is in the stable phase or clearing phase. Note that the "surrogate" *legStatus* for the controlling leg indicates the charging relationship between the involved passive legs after the call has been transferred.

The characteristics of the "Stable_Multi_Passive_Party" CSCV state follow:

a) Relationship with the BCSM:

> controlling *leg*:
> *legStatus* = surrogate
>
> passive *leg* = p1 to pn:
> *legStatus* = joined

CSCV state "Stable_Multi_Passive_Party" encompasses the following BCSM PICs and DPs:

O_BCSM leg p1:
DPs: O_Term_Seized DP, O_Answer DP, O_Suspend DP, O_Re-answer DP, O_Mid_Call DP {O_Active}, O_Mid_Call DP {O_Suspended}.
If more than 2 legs: O_Disconnect DP, O_No_Reply DP, O_Called_Party_Busy DP.
PICs: O_Alerting, O_Active, O_Suspended,

T_BCSM leg p1:
DPs: T-Abandon DP, T_Mid_Call {T_Active} DP
If more than 2 legs: T-Disconnect DP,
PICs: Present_Call, T_Alerting, T_Active,

O_BCSM (leg p2 to Pn) "stable passive party leg":
DPs: O_Term_Seized DP for leg p2, O_Answer DP for leg p2, O_Suspend DP for leg p2, O_Re-answer DP for leg p2, O_Mid_Call DP {O_Active} for leg p2, O_Mid_Call DP (O_Suspended} for leg p2,
If more than 2 legs: O_Disconnect DP, O_No_Answer DP, O_Called_Party_Busy DP.
PICs: O_Alerting, O_Active, O_Suspended.

---

b) Entry Events:

See the exit events for the CSCV states:

- "Stable_Multi_Party".

- "Stable_1_Party".

- "Forward".

c) Exit Events:

Summary of CSCV state-related SSF events:

O_Called_Party_Busy DP for leg p, O_No_Answer DP for leg p, O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, O_Disconnect DP for leg p, O_Mid_Call DP {O_Active} for leg p, O_Mid_Call DP {O_Suspended} for leg p, T_Suspend DP for leg p, T_Re-Answer DP for leg p, T_Mid_Call {T_Active} DP for leg p, T_Abandon for leg p and T_Disconnect DP for leg p.

Direct Transitions from the "Stable_Multi_Passive_Party" CSCV state:

i) To the deletion of the "Stable_Multi_Passive_Party" CSCV state (deletion of CS instance)

-   Processing of a valid ReleaseCall operation received from SCF, in response to any DP that is allowed to be reported in the "Stable_Multi_Passive_Party" CSCV state.

-   Processing of a valid ReleaseCall operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

ii) To the "Forward" CSCV state

- Detection of the O_Disconnect DP for leg p sent from the O_BCSM provided the legs before the release in the "Stable_Multi_Passive_Party" CSCV state were equal to 2.

- Detection of the O_No_Answer DP for leg p sent from the O_BCSM on no-reply timer expiry or receipt of a release event from the remote T_BCSM, provided the legs before the release in the "Stable_Multi_Passive_Party" CSCV state were equal to 2.

- Detection of the O_Called_Party_Busy DP for leg p sent from the O_BCSM, provided the legs before the release in the "Stable_Multi_Passive_Party" CSCV state were equal to 2.

iii) To the "Stable_Multi_Passive_Party" CSCV state

- Detection of T_Disconnect DP for leg p (if leg p1 associated with T_BCSM), provided the passive legs before the processing of the DP event in the "Stable_Multi_Passive_Party" CSCV state were greater than 2 (see notes 2 and 3).

- Detection of the O_Disconnect DP for leg p sent from the O_BCSM, provided the passive legs before the release in the "Stable_Multi_Passive_Party" CSCV state were greater than 2 (see notes 2 and 3).

- Detection of the O_No_Answer DP for leg p sent from the O_BCSM on expiry of no-reply timer or on receipt of a release event from the remote T_BCSM, provided the passive legs before the release in the "Stable_Multi_Passive_Party" CSCV state were greater than 2 (see notes 2 and 3).

- Detection of the O_Called_Party_Busy DP for leg p sent from the O_BCSM, provided the passive legs before the release in the "Stable_Multi_Passive_Party" CSCV state were greater than 2 (see notes 2 and 3).

- Detection of the O_Mid_Call DP {O_Active}, O_Mid_Call DP {O_Suspended}.for leg p sent from the O_BCSM on receipt of a FeatureIndReqInd signalling event from the remote T_BCSM or detection of a T_Mid_Call DP{T_Active} for leg p sent from the T-BCSM or receipt of a FeatureIndReqInd signalling event from the remote O_BCSM.

- Processing of a valid DisconnectLeg operation for *leg* p received from SCF, in response to an O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p,; O_Mid_Call DP {O_Active}for leg p, O_Mid_Call DP {O_Suspended}.for leg p
O-Disconnect DP for leg p, O_No_Reply DP for leg p, O_Called_Party_Busy DP for leg p and T_Mid_Call {T_Active} DP for leg p, T_Disconnect DP for leg p, T-Abandon DP for leg p provided the passive legs before the processing of the DisconnectLeg(p) operation in the "Stable_Multi_Passive_Party" CSCV state were greater than 2 (see note 2).

- Processing of a valid DisconnectLeg(p) operation for leg p received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA), provided the legs before the processing of the DisconnectLeg(p) operation in the "Stable_Multi_Passive_Party" CSCV state were greater than 2 (see note 2).

- Processing of a ExportLeg(p) due to a valid MoveLeg(p) or SplitLeg(p) operation received from SCF, in response to an O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, O_Mid_Call DP {O_Active}for leg p, O_Mid_Call DP {O_Suspended}.for leg p, O-Disconnect DP for leg p, O_No_Reply DP for leg p, O_Called_Party_Busy DP for leg p and T_Mid_Call {T_Active} DP for leg p, T_Disconnect DP for leg p, T-Abandon DP for leg p provided the legs before the processing of the operation in the "Stable_Multi_Passive_Party" CSCV state were greater than 2.

- Processing of a ExportLeg(p) due to a valid MoveLeg(p) or SplitLeg(p) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA), provided the legs before the processing of the operation in the "Stable_Multi_Passive_Party" CSCV state were greater than 2.

- Processing of a ImportLeg(p) due to a valid MoveLeg(p) or MergeCallSegments (target CS) operation received from SCF, in response to an O_Term_Seized DP for leg p O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, O_Mid_Call DP {O_Active}for leg p, O_Mid_Call DP {O_Suspended}.for leg p, O-Disconnect DP for leg p, O_No_Reply DP for leg p, O_Called_Party_Busy DP for leg p and T_Mid_Call {T_Active} DP for leg p, T_Disconnect DP for leg p, T_Abandon DP for leg p.

- Processing of a ImportLeg(p) due to a valid MoveLeg(p) or MergeCallSegments (target CS) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a valid Continue or ContinueWithArgument operation received from SCF, in response to an O_Disconnect DP for leg c, O_Abandon for leg c, T_Disconnect for leg c, T_No_Answer for leg c, T_Busy for leg c provided the number of passive joined legs before resumption is greater than 1 (see notes 1 and 2).

NOTE 1:   When a release event is detected on the c-leg in the CS, the leg state is changed (from "joined" to "surrogate").

NOTE 2:   The resumption of call processing (Continue or ContinueWithArgument) following the release event reported for the controlling leg, will not cause a propagation of the release event, all passive legs are retained (because the number of remaining joined legs in the CS is greater than 1).

NOTE 3:   As a result of a release event on a passive leg a transient CSCV state (not modelled) "multi-passive party-pending" exists before reaching either the "Stable 1-Party" or the "Stable Multi-Passive Party" CSCV state. The DisconnectLeg(p) operation is accepted in this implicit "multi-passive-party-pending" CSCV state.

NOTE 4:   Will remain in the CSCV state for any valid DP and PIC that comprises the definition of this CSCV.

iv) To the "Stable_1_Party" CSCV state

- Detection of a T_Disconnect DP for leg p (if leg p is associated with a T_BCSM), on receipt of a ReleaseReqInd signalling event from the remote O_BCSM, provided the passive legs before the processing of the DP event in the "Stable_Multi_Passive_Party" CSCV state were equal to 2 (see note 1).
  Processing of a valid DisconnectLeg operation for *leg* p received from SCF, in response to an O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, O_Mid_Call DP {O_Active}for leg p, O_Mid_Call DP (O_Suspended).for leg p, and T_Mid_Call {T_Active} DP for leg p, T_Disconnect DP for leg p, T-Abandon DP for leg p, provided the passive legs before the processing of the DisconnectLeg(p) operation in the "Stable_Multi_Passive_Party" CSCV state were equal to 2.

- Processing of a valid DisconnectLeg(p) operation for leg p received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA), provided the passive legs before the processing of the DisconnectLeg(p) operation in the "Stable_Multi_Passive_Party" CSCV state were equal to 2.

- Processing of a ExportLeg(p) due to a valid MoveLeg(p) or SplitLeg(p) operation received from SCF, in response to an O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, O_Mid_Call DP {Active}for leg p, O_Mid_Call DP {O_Suspended}.for leg p and T_Mid_Call {Active} DP for leg p, T_Disconnect DP for leg p, T-Abandon DP for leg p provided the passive legs before the processing of the operation in the "Stable_Multi_Passive_Party" CSCV state were equal to 2.

- Processing of a ExportLeg(p) due to a valid MoveLeg(p) or SplitLeg(p) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA), provided the passive legs before the processing of the operation in the "Stable_Multi_Passive_Party" CSCV state were equal to 2.

NOTE 5:   As a result of a release event on a passive leg a transient CSCV state (not modelled) "multi-passive party-pending" exists before reaching either the "Stable 1-Party" or the "Stable Multi -Passive Party" CSCV state. The DisconnectLeg(p) operation is accepted in this implicit "multi-passive-party-pending" CSCV state.

v) To the "Stable_Multi_Party" CSCV state

The following procedures have the import of the controlling leg in the "Stable_Multi_Passive_Party" CSCV state when, in the CSA to which this CS belongs, first a CS with the controlling leg joined is imported using the MoveCallSegments operation, then the MoveLeg or MergeCallSegments operation is used to move the controlling leg into this CS.

- Processing of a ImportLeg(c) due to a valid MoveLeg(c) or MergeCallSegments (as Target CS) operation received from SCF, in response to an O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, O_Mid_Call DP {Active}for leg p, O_Mid_Call DP {O_Suspended}.for leg p, T_Mid_Call {Active} DP for leg p, T_Disconnect DP for leg p, T-Abandon DP for leg p.

- Processing of a ImportLeg(c) due to a valid MoveLeg(c) or MergeCallSegments (as Target CS) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

6.6.3.4.3.10   "Stable_Multi_Party" Call Segment Connection View State

The "Stable_Multi_Party" CSCV state represents a stable or clearing multi-party call composed by 3 or more legs in one call segment.

The characteristics of the "Stable_Multi_Party" CSCV state follow:

a) Relationship with the BCSM:

> controlling leg = c:
> legStatus = joined;
>
> passive leg = p1 to pn:
> legStatus = joined.

CSCV state "Stable_Multi_Party" encompasses the following BCSM PICs and DPs:

O_BCSM:,
DPs leg p: O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Mid_Call DP {O_Active} for leg p, O_Mid_Call DP (O_Suspended} for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p
DPs leg c: O_Mid_Call DP{ O_Alerting } for leg c, O_Mid_Call DP {O_Active} for leg c, O_Mid_Call DP (O_Suspended} for leg c.
PICs: O_Alerting, O_Active, O_Suspended,

T_BCSM:
DPs leg c: Call_Accepted DP for leg c, T_Answer DP for leg c, T_Suspend DP for leg c, T_Re_Answer DP for leg c, T_Mid_Call DP {T_Active} for leg p.
DPs leg p: T_Mid_Call DP {T_Active} for leg c.
PICs: T_Alerting, T_Active, T_Suspended.

---

b) *Entry Events*:

See the exit events for the CSCV states:

- "Terminating_Setup".

- "Stable_2_Party".

- "Stable_Multi_Passive_Party".

c) Exit Events:

Summary of CSCV state-related SSF events:

Switch_Hook_Flash_Immediate trigger, BRI_Feature_Activation_Indicator trigger, second dial tone associated with the Switch_Hook_Flash_Specified_Code trigger, O_Mid_Call DP{ O_Alerting,} for leg c, O_Mid_Call DP {O_Active,O_Suspended} for leg c and leg p, O_Abandon DP for leg c, O_Called_Party_Busy DP for leg p, O_No_Answer DP for leg p, O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, O_Disconnect DP for leg c and leg p, T_Suspend DP for leg c, T_Re-answer DP for leg c T_Mid_Call DP {T_Active} for leg c and leg p and T_Disconnect DP for leg c and leg p, T_Abandon for leg p.

Direct Transitions from the "Stable_Multi_Party" CSCV state:

i) To the deletion of the "Stable_Multi_Party" CSCV state (deletion of CS instance)

- Processing of a valid ReleaseCall operation received from SCF, in response to any DP that is allowed to be reported in the Stable_Multi_Party CSCV state.

- Processing of a valid ReleaseCall operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

ii) To the "Stable_Multi_Party" CSCV state

- Detection of an O_Called_Party_Busy DP for leg p, O_No_Answer DP for leg p, O_Disconnect DP leg p, T_Abandon for leg p, T_Disconnect DP for leg p, provided the remaining number of passive legs are greater than 2 (see note 3).

- Processing of a valid DisconnectLeg operation for *leg* p received from SCF, provided the remaining passive legs are greater than 2, in response to:

  - an OMidCall or TMidCall DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or O_Mid_Call DP{ O_Alerting, for leg c, O_Mid_Call DP{ O_Active} for leg c and leg p, O_Mid_Call DP{ O_Suspended} for leg c and leg p or T_Mid_Call DP {T_Active} for leg c and leg p);

  - O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a valid DisconnectLeg(p) operation for leg p received from SCF which was initiated by the SCF, provided the remaining passive legs are greater than 1 (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a ExportLeg(p) due to a valid MoveLeg(p) or SplitLeg(p) operation received from SCF, provided the remaining passive legs are greater than 1, in response to:

  - an OMidCall or TMidCall DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or O_Mid_Call DP{ O_Alerting, } for leg c, O_Mid_Call DP{ O_Active} for leg c and leg p, O_Mid_Call DP{ O_Suspended} for leg c and leg p or T_Mid_Call DP {T_Active} for leg c and leg p);

  - O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a ExportLeg(p) due to a valid MoveLeg(p) or SplitLeg(p) operation received from SCF which was initiated by the SCF, provided the remaining passive legs are greater than 1, (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a ImportLeg(p) due to a valid MoveLeg(p) or MergeCallSegments (as Target CS) operation received from SCF, in response to:

  ▪ an OMidCall or TMidCall DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or O_Mid_Call DP{ O_Alerting, } for leg c, O_Mid_Call DP{ O_Active} for leg c and leg p, O_Mid_Call DP{ O_Suspended} for leg c and leg p or T_Mid_Call DP {T_Active} for leg c and leg p);

  ▪ O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a ImportLeg(p) due to a valid MoveLeg(p) or MergeCallSegments (as Target CS) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a valid ContinueWithArgument operation in response to, O_Called_Party_Busy DP for leg p, O_No_Answer DP for leg p, O_Disconnect DP for leg p and T_Disconnect DP for leg p,, T-Abandon DP for leg p provided the remaining passive legs are greater than 1 (see note 2).

NOTE 1: Will remain in the CSCV state for any valid DP and PIC that comprises the definition of this CSCV.

NOTE 2: The Continue operation is not allowed for a single call segment CSA with more than two legs or a multi call segment CSA. In this case an error procedure is invoked.

NOTE 3: As a result of a release event on a passive leg a transient CSCV state (not modelled) "Multi-party-pending" exists before reaching either the "Stable 2-Party" or the "Stable Multi Party" CSCV state.
        The DisconnectLeg(p) operation is accepted in this implicit "Multi-party-pending" CSCV state.

iii) To the "Stable_2_Party" CSCV state

- Detection of an O_Called_Party_Busy DP for leg p, O_No_Answer DP for leg p, O_Disconnect DP leg p, T_Abandon for leg p, T_Disconnect DP for leg p, provided the remaining legs are equal to 2 (see note 4).

- Processing of a valid DisconnectLeg operation for *leg* p received from SCF, provided the remaining legs are equal to 2, in response to:

  ▪ an OMidCall or TMidCall DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or O_Mid_Call DP{ O_Alerting, } for leg c, O_Mid_Call DP{ O_Active} for leg c and leg p, O_Mid_Call DP{ O_Suspended} for leg c and leg p, or T_Mid_Call DP {T_Active} for leg c and leg p);

  ▪ O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a valid DisconnectLeg(p) operation for leg p received from SCF which was initiated by the SCF, provided the remaining legs are equal to 2 (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Processing of a ExportLeg(p) due to a valid MoveLeg(p) or SplitLeg(p) operation received from SCF, provided the remaining legs are equal to 2, in response to:

  ▪ an OMidCall or TMidCall DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or O_Mid_Call DP{ O_Alerting, } for leg c, O_Mid_Call DP{ O_Active} for leg c and leg p, O_Mid_Call DP{ O_Suspended} for leg c and leg p or T_Mid_Call DP {T_Active} for leg c and leg p);

  ▪ O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, T_Suspend DP for leg c, T_Re-answer DP for leg c.

Processing of a ExportLeg(p) due to a valid MoveLeg(p) or SplitLeg(p) operation received from SCF which was initiated by the SCF, provided the remaining legs are equal to 2, (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

NOTE 4:   As a result of a release event on a passive leg a transient CSCV state (not modelled) "Multi-party-pending" exists before reaching either the "Stable 2-Party" or the "Stable Multi Party" CSCV state. The DisconnectLeg(p) operation is accepted in this implicit "Multi-party-pending" CSCV state.

iv) To the "Stable_Multi_Passive_Party" CSCV state

- Processing of a valid DisconnectLeg operation for *leg* c received from SCF, in response to:

  ▪ an OMidCall or TMidCall DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or O_Mid_Call DP{O_Alerting, } for leg c, O_Mid_Call DP{ O_Active} for leg c and leg p, O_Mid_Call DP{ O_Suspended} for leg c and leg p or T_Mid_Call DP {T_Active} for leg c and leg p);

  ▪ O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a valid DisconnectLeg operation for leg c received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

- Detection of a release event for leg c, e.g. O_Abandon DP for leg c, O_Disconnect DP for leg c, T_No_Answer DP for leg c, T_Busy DP for leg c and T_Disconnect DP for leg c (see note 5).

NOTE 5:   A release event received on the joined controlling leg causes the leg state to become surrogate. Notice that in this case the resumption of call processing (ContinueWithArgument) following the release event reported for the controlling leg, will not cause a propagation of the release event, all passive legs are retained if the number of remaining joined legs in the CS is greater than 1.

- Processing of a ExportLeg(c) due to a valid MoveLeg(c) or SplitLeg(c) operation received from SCF, in response to:

  ▪ an OMidCall or TMidCall DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or O_Mid_Call DP{ O_Alerting, } for leg c, O_Mid_Call DP{ O_Active} for leg c and leg p, O_Mid_Call DP{ O_Suspended} for leg c and leg p or T_Mid_Call DP {T_Active} for leg c and leg p);

  ▪ O_Term_Seized DP for leg p, O_Answer DP for leg p, O_Suspend DP for leg p, O_Re-answer DP for leg p, T_Suspend DP for leg c, T_Re-answer DP for leg c.

- Processing of a ExportLeg(c) due to a valid MoveLeg(c) or SplitLeg(c) operation received from SCF which was initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision, an user to service information or any other valid DP or notification/event received from an associated CS of the same CSA).

6.6.3.5   Examples of CV Configurations, Composite Transitions

6.6.3.5.1   Examples of Composite Transitions from the "Originating_Setup" CSCV state

i) To the "Originating_1_Party_Setup" CSCV state of a new Call Segment in the same CSA

- Processing of a valid InitiateCallAttempt operation received from SCF, in response to an Origination_Attempt_Authorized DP for leg c, Collected_Information DP for leg c, or Analysed_Information DP for leg c. In the same CSA a new call segment is created and moved into the "Originating_1_Party_Setup" CSCV state.

6.6.3.5.2   Examples of Composite Transitions from the "Stable_2_Party" CSCV state

i) To the "Originating_Setup" CSCV state

- Processing of a valid message containing DisconnectLeg operation for *leg* p1, plus a Connect operation received from SCF, in response to an OMidCall DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or a mid-call EDP).

- In response to an OMidCall or TMidCall DP, processing of a valid SplitLeg operation, followed by a CreateCallSegmentAssociation operation (which creates a null CSA), followed by a MoveCallSegments operation (to move CS 2 from its current location in the source CSA into a new target CSA) and a CollectInformation operation on the CS which remains in the source CSA (to transition the unmoved CS into the "Originating Setup" CSCV state).

NOTE 1: For T_MidCall DP, this transition is valid at a local exchange and not at a transit exchange.

- Processing of a valid message containing DisconnectLeg for *leg* p1, plus Connect operation received from SCF, initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision or an user to service information).

- Processing of a valid message containing DisconnectLeg for *leg* p1, plus Connect operation received from SCF, in response to an O_Suspended DP for *leg* p1.

- Processing of a valid Connect operation received from SCF, in response to an O_Disconnect DP for *leg* p1.

ii) To the "Stable_1_Party" CSCV state

- Processing of a valid SplitLeg operation for leg c or leg p received from SCF, followed by user interaction, and then a valid Connect operation received from SCF, in response to an OMidCall DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or a mid-call EDP).

NOTE 2: The user interaction and then the Connect operation are performed for the:
- created CS for the SplitLeg(c) operation; and
- initial CS for the SplitLeg(p) operation.

- Processing of a valid SplitLeg operation received from SCF, followed by user interaction, and then a valid Connect operation received from SCF, in response to a TMidCall DP (reporting the detection of a Switch_Hook_Flash_Immediate or BRI_Feature_Activation_Indicator trigger, or a mid-call EDP).

NOTE 3: The user interaction and then the Connect operation are performed for the:
- created CS for the SplitLeg(c) operation; and
- initial CS for the SplitLeg(p) operation.

- Processing of a valid SplitLeg operation received from SCF, which was initiated by the SCF (caused by, e.g. an user to service information), followed by user interaction, and then a valid Connect operation received from SCF.

NOTE 4: The user interaction and then the Connect operation are performed for the:
- created CS for the SplitLeg(c) operation; and
- initial CS for the SplitLeg(p) operation.

- The detection of a "flash", followed by the application of second dial tone for a Switch_Hook_Flash_Specified_Code trigger.

iii) To the "Forward" CSCV state

- In response to an OMidCall or TMidCall DP, processing of a valid SplitLeg operation for leg p, followed by a CreateCallSegmentAssociation operation (which creates a null CSA), followed by a MoveCallSegments operation (to move CS 2 from its current location in the source CSA into a new target CSA) and a Connect operation on the moved CS in the target CSA (to forward the call).

- Processing of a valid Connect operation received from SCF, in response to a T_Disconnect DP for *leg* c.

iv) To the "Stable 2-party" CSCV state

- In response to an OMidCall or TMidCall DP, processing of a valid SplitLeg operation for leg p, followed by a CreateCallSegmentAssocation operation (which creates a null CSA), followed by a MoveCallSegments operation (to move CS 2 from its current location in the source CSA into a new target CSA) and a Connect operation on the CS which remains in the source CSA (to transition the unmoved CS back into the Stable_2_party CSCV state).

NOTE 5: For TMidCall DP, this transition is valid at a local exchange and not at a transit exchange.

6.6.3.5.3   Examples of Composite Transitions from the "Stable_Multi_Passive_Party" CSCV state provided the CS has only 2 pending legs

i) To the "Stable_2_Party" CSCV state

- In response to an O_Disconnect or T_Disconnect DP for *leg pn*, processing of a valid MoveCallSegments operation (to move the CS in a "Stable_Multi_Passive_Party" CSCV state from a source CSA to a target CSA), followed by a valid MergeCallSegments operation on the target CSA (to merge two CSs in the target CSA into a "Stable_2_Party" CSCV state).

- Processing of a valid DisconnectLeg operation for *leg* pn for the CS in a "Stable_Multi_Passive_Party" CSCV state, followed by a valid MoveCallSegments operation (to move the CS in a "Stable_Multi_Passive_Party" CSCV state from a source CSA to a target CSA), followed by a valid MergeCallSegments operation on the target CSA (to merge two CSs in the target CSA into a "Stable_2_Party" CSCV state), initiated by the SCF.

ii) To the "Forward" CSCV state

- Processing of a valid message containing DisconnectLeg for *leg* p2, plus Connect received from SCF, in response to an O_Suspended DP for *leg* p2.

- Processing of a valid Connect operation received from SCF, in response to an O_Disconnect DP for *leg* p2.

- Processing of a valid Connect operation received from SCF, in response to a T_Disconnect DP for *leg* p2.

- Processing of a valid message containing DisconnectLeg for *leg* p2, Connect received from SCF, initiated by the SCF (caused by, e.g. a charging notification from the SSF, an SCF time supervision or an user to service information).

6.6.3.6   Overview of Transitions of DP Events to the CSCV states

Table 17 and CSCV state diagrams provide an overview of the transitions of DP events to the CSCV states.

Table 17: Transition of DP Event to CSCV state

CSCV State → Detection of DP event↓	Originating _Setup	Originating_ 1_Party _Setup	Stable_1 _Party	Terminating _Setup	1_Party	Stable_2_ Party	Forward	Stable_ Multi_ Passive_ Party	Stable_ Multi_Party
Origination_ Attempt DP for leg c	**Originating _Setup** note 5	NA: note 6	NA	NA	**NA**	NA	NA	NA	NA
Origination_ Attempt_ Denied DP for leg c and leg p {due to InitiateCallAttempt or Connect for call diversion)	**Originating _Setup**	NA note 6	NA	NA	NA	NA	**Forward** for leg p2	NA	NA
Origination_ Attempt_ Authorized DP for leg c and leg p {due to InitiateCallAttempt or Connect for call diversion)	*Originating _Setup*	*Originating_ 1_Party_ Setup*	NA	NA	NA	NA	**Forward** for leg p2	NA	NA
Collected_ Information DP for leg c and leg p {due to InitiateCall-Attempt or Connect for call diversion)	**Originating _Setup**	**Originating_ 1_Party_ Setup**	NA	NA	NA	NA	**Forward** for leg p2	NA	NA

CSCV State → / Detection of DP event↓	Originating_Setup	Originating_1_Party_Setup	Stable_1_Party	Terminating_Setup	1_Party	Stable_2_Party	Forward	Stable_Multi_Passive_Party	Stable_Multi_Party
Analysed_Information DP for leg c and leg p {due to InitiateCallAttempt or Connect and SelectFacility (with destination addr) for call diversion)	*Originating_Setup* *for leg c*	*Originating_1_Party_Setup* *for leg p*	NA	NA	*NA*	NA	**Forward** for leg p2	NA	NA
O_Term_Seized DP for leg p	**Stable_2_Party**	**Stable_1_Party**	NA	NA	NA	NA	**Stable_Multi_Passive_Party** for leg p2	**Stable_Multi_Passive_Party** for leg p2	*NA*
Route_Select_Failure DP for leg p	*Originating_Setup*	*Originating_1_Party_Setup*	NA	NA	NA	**Originating_Setup**	*Forward* *for leg p2*	NA	NA
Authorized_Route_Failure DP for leg p	*Originating_Setup*	*Originating_1_Party_Setup*	NA	NA	NA	NA	**Forward** for leg p2	NA	NA
O_Called_Party_Busy DP for leg p	**Originating_Setup**	**Originating_1_Party_Setup**	Originating_1_Party_Setup	NA	NA	Originating_Setup	**Forward** for leg p2	If passive legs = 2 before release **Forward** else if > 2 then **Stable_Multi_Passive_Party**	If remaining legs = 2 **Stable_2_Party** else **Stable_Multi_Party**

ETSI

CSCV State → Detection of DP event↓	Originating _Setup	Originating_ 1_Party _Setup	Stable_1 _Party	Terminating _Setup	1_Party	Stable_2_ Party	Forward	Stable_ Multi_ Passive_ Party	Stable_ Multi_Party
O_No_Answer DP for leg p	**Originating _Setup**	**Originating_ 1_Party _Setup**	Originating_1 _Party_ Setup	NA	NA	Originating_ Setup	**Forward** for leg p2	If passive legs = 2 before release **Forward** else if more than 2 legs then **Stable_ Multi_ Passive_ Party**	If remaining legs = 1 **Stable_2_ Party** else **Stable_ Multi_Party**
O_Answer DP for leg p	**Stable_2_ Party**	**Stable_1_ Party**	**Stable_1_ Party**	NA	NA	Stable_2_ Party	NA	Stable_ Multi_ Passive_ Party	**Stable Multi_Party**
O_Suspend DP for leg p	NA	NA	***Stable_1_ Party***	NA	NA	Stable_2_ Party	NA	Stable_ Multi_ Passive_ Party	Stable Multi_Party
O_Re-answer DP for leg p	NA	NA	**Stable_1_ Party**	NA	NA	Stable_2_ Party	NA	Stable_ Multi_ Passive_ Party	Stable Multi_Party
O_Mid_Call DP {Send_Call PIC} for leg c	**Originating _Setup**	NA	NA	NA	NA	**NA**	NA	NA	NA
O_Mid_Call DP {O_Alerting PIC} for leg c	NA	NA	**Stable_1_ Party**	NA	1_Party	Stable_2_ Party	NA	NA	**Stable Multi_Party**
O_Mid_Call DP {O_Active PIC} for leg c and leg p	NA	NA	***Stable_1_ Party***	NA	NA	Stable_2_ Party	NA	Stable_ Multi_ Passive_ Party	Stable Multi_Party
O_Mid_Call DP {O_Suspended PIC}for leg c and leg p	NA	NA	***Stable_1_ Party***	NA	NA	**Stable_2_ Party**	NA	Stable_ Multi_ Passive_ Party	Stable Multi_Party

ETSI

CSCV State → / Detection of DP event↓	Originating_ Setup	Originating_1_Party_Setup	Stable_1_Party	Terminating_Setup	1_Party	Stable_2_Party	Forward	Stable_Multi_Passive_Party	Stable_Multi_Party
O_Disconnect DP for leg c and leg p	NA	NA	For leg p: **Originating_1_Party_Setup** For leg c: NA	NA	**CS Instance deleted** note 4	For leg p: **Originating_Setup** For leg c: **Stable_1_Party**	For leg p1: CS instance deleted else NA	For leg p: If passive legs = 2 before release **Forward** else if more than 2 legs Stable_Multi_Passive_Party For leg c: NA	For leg p: If remaining legs = 1 **Stable_2_Party** else **Stable_Multi_Party** For leg c: Stable_Multi_Passive_Party
O_Abandon DP for leg c	CS instance deleted	NA	*NA*	NA	CS instance deleted	Stable_1_Party	NA	NA	**Stable_Multi_Passive_Party**
Terminating_Attempt_ DP for leg p	NA	NA	NA	**Terminating_Setup** note 5:	NA	NA	NA	NA	NA
Terminating_Attempt_Authorized DP for leg p	NA	NA	NA	Terminating_Setup	NA	NA	NA	NA	NA
Terminating_Attempt_Denied DP for leg p	NA	NA	NA	**Terminating_Setup**	NA	NA	NA	NA	NA
Facility_Selected_and_Availlable DP for leg c	NA	NA	*NA*	*Terminating_Setup*	NA	NA	NA	NA	NA
Call_Accepted DP for leg C	NA	NA	*NA*	*Stable_2_Party*	NA	NA	NA	NA	NA
T_Busy DP for leg c	NA	NA	NA	*Terminating_Setup*	**CS Instance deleted** note 4	**Terminating_Setup**	NA	NA	**Stable_Multi_Passive_Party**

CSCV State → Detection of DP event↓	Originating _Setup	Originating_ 1_Party _Setup	Stable_1 _Party	Terminating _Setup	1_Party	Stable_2_ Party	Forward	Stable_ Multi_ Passive_ Party	Stable_ Multi_Party
T_No_Answer DP for leg c	NA	NA	NA	*Terminating _Setup*	**CS Instance deleted** note 4	**Termina ting_ Setup**	NA	NA	**Stable_ Multi_ Passive_ Party**
T_Answer.DP for leg c	NA	NA	**NA**	Stable_2_ Party	1-Party	Stable_2_ Party	NA	NA	*Stable_ Multi_Party*
CSCV State → Detection of DP event↓	Originating _Setup	Originating_ 1_Party _Setup	Stable_1 _Party	Terminating _Setup	1_Party	Stable_2_ Party	Forward	Stable_ Multi_ Passive_ Party	Stable_ Multi_Party
T_Suspend DP for leg c	NA	NA	*NA*	NA	**1-Party**	*Stable_2_P arty*	NA	NA	*Stable_ Multi_Party*
T_Re-answer.DP for leg c	NA	NA	**NA**	NA	**1_Party**	Stable_2_ Party	NA	NA	**Stable_ Multi_Party**
T_Midcall DP{T_Active} for leg c and leg p	NA	NA	For leg p: **Stable_1_ Party** For leg c: NA	*NA*	For leg c: **1_Party** For leg p NA	**Stable_2_ Party**	For leg p: **Forward** For leg c: NA	For leg p: **Stable_ Multi_ Passive_ Party** For leg c: NA	Stable_ Multi_Party
T_Disconnect DP for leg c and leg p	NA	NA	**CS Instance deleted** note 4	NA	**CS Instance deleted** note 4	For leg p: **1_Party** For leg c: **Stable_1_ Party**	For leg p1: **CS instance deleted** else NA	For leg p: If passive legs = 2 before release **Stable_1_ Party** else if > 2 then **Stable_ Multi_ Passive_ Party** For leg c: NA	For leg p: If remaining passive legs = 1 **Stable_2_ Party** else **Stable_ Multi_Party** For leg c: **Stable_ Multi_ Passive_ Party**

CSCV State → Detection of DP event↓	Originating_ Setup	Originating_ 1_Party _Setup	Stable_1 _Party	Terminating _Setup	1_Party	Stable_2_ Party	Forward	Stable_ Multi_ Passive_ Party	Stable_ Multi_Party
T_Abandon for leg p	NA	NA	**CS Instance deleted** note 4	CS instance deleted	NA	1_Party	For leg p1: **CS instance deleted** else NA	NA	If remaining passive legs = 1 **Stable_ 2_Party** else **Stable_ Multi_Party**

NOTE 1: The SLPI in the SCF becomes aware of the CS transition into Stable_2_Party/Stable_1_Party by the reporting of the O_Term_Seized DP/O_Answer DP in case of an O_BCSM. When the SCF sends a CPH operation before having received one of these DP notification, the operation will be processed as an error.
NOTE 2: The SLPI in the SCF becomes aware of the CS transition into Stable_2_Party by the reporting of the Call_Accepted DP/T_Answer DP in case of an T_BCSM.
NOTE 3: In order to secure a correct connection view for service execution, the SLPI in the SCF becomes updated about the CSCV state evolution in the SSF via DP arming and DP reporting.
NOTE 4: If the DP is armed as an EDP it is reported to the SCF together with any other pending reports (ApplyChargingReport, CallInformationReport). The CS instance will subsequent be deleted. This occurs immediately in case call processing was not suspended otherwise (DP armed as EDP-R) upon resumption of the suspended call.
NOTE 5: DP cannot occur as an EDP only as a TDP.
NOTE 6: With InitiateCallAttenpt the O_BCSM becomes suspended at DP Origination_Attempt_Authorized.
NOTE 7: Detection of a Route_Select_Failure DP after a Release event which generated a "Route_Failure" event causing a transition from PIC Alerting to PIC Select_Route.

Abbreviation used:
NA = Not Applicable
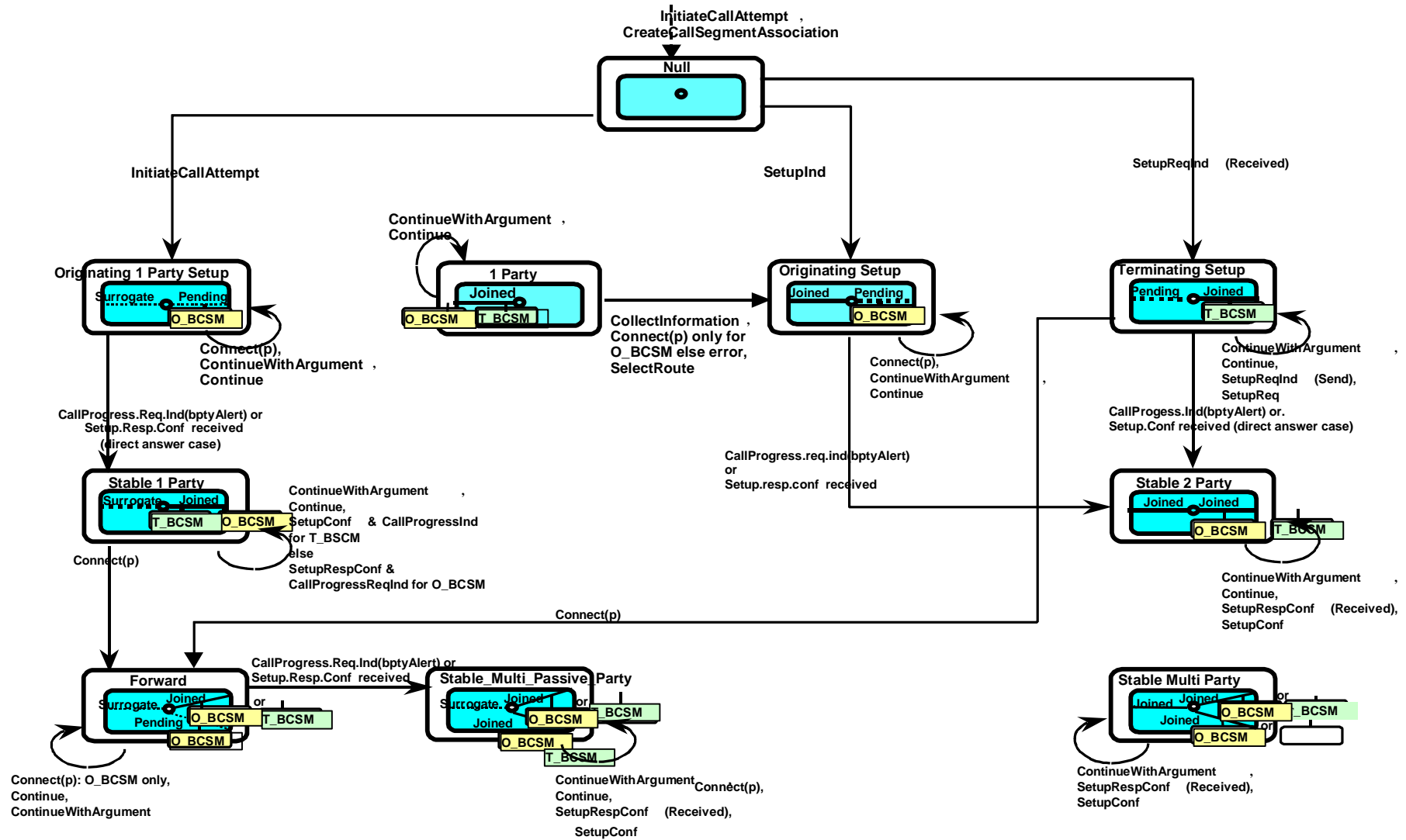
Figure 33: Overview 1: CSCV Transitions
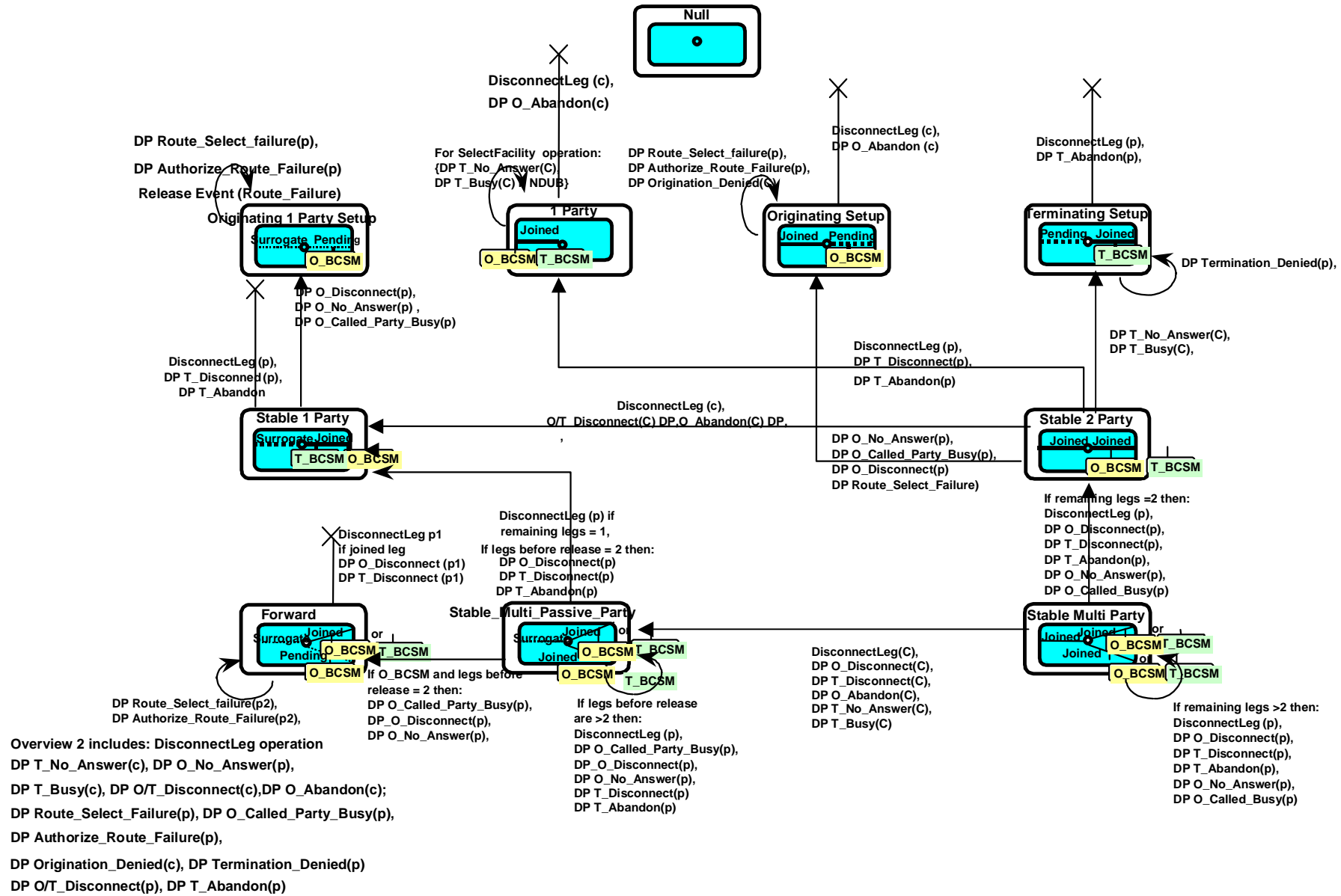
Figure 34: Overview 2: CSCV Transitions

6.7   Out-Channel Call-Related User Interaction (OCCRUI)

6.7.1   Description of the OCCRUI at the CCF/SSF "Call Model" level

The Out-channel call related user interaction provides the ability to communicate information within the context of a call on the out-channel signalling access (within the same call reference of a call).

A generic transport mechanism (transparent at the SSF level) for the exchange of information between the "User" and the "Service Logic" is supported based on INAP operations, respectively SendSTUI in the "SCF-to-User" direction and ReportUTSI in the "User to SCF" direction. These user information (USI) is composed of two elements (parameters) which are USIServiceIndicator and USIInformation. The first parameter identifies the IN Service Logic/Service Feature invoked while the second carries useful information between the User/Service Logic and the Service Logic. Direct User-to-User signalling using the OCCRUI mechanism is prohibited.

> NOTE: Refer to EN 301 931-1 for a definition of the term "User" in the context of the OCCRUI mechanism.

A *User to Service Information (USI)* refers to either a UTSI or a STUI.

In the "SCF-to-User" direction, once the SSF receives an STUI from the SCF within the sendSTUI operation, it forwards it to the appropriate user application in the network, e.g. an ISDN User. This User is clearly and easily defined by the Call Reference (indicated by the down-lower protocols) and the legID parameter (indicated by the INAP protocol).

In the "User-to-SCF" direction the CCF/SSF decides whether it forwards the USI information it receives from a User to the succeeding/preceding exchange or it passes it to a specific IN service. Two cases applies:

1) The USI information is considered as a "notification event" previously requested by the SCF. In this case, there is already a SSF-SCF relationship.

2) The USI information is an additional information. In this case, it is only an optional information.

The SSF addresses the appropriate SCF based upon the *ServiceIndicator* parameter it receives within the USI. The *ServiceIndicator* value is indicated by the User (e.g. ISDN User) or by the Service Logic:

- If the SCF has initiated the USI dialogue, the User sets the *ServiceIndicator* value of the USI to the *ServiceIndicator* value as requested by the SCF. This scenario corresponds to the first case.

- If the user sends an USI without having previously received an STUI from the SCF, then it initializes the *ServiceIndicator* value of the USI with an predefined value. This scenario corresponds to the second case.

6.7.1.1   First case: the USI information is considered as a "notification previously requested by the SCF"

In this case, the SSF communicates with the SCF during an already existing SCF-SSF relationship; the SCF initiates the "Out-Channel" dialogue with the User sending an STUI, within the SendSTUI operation. Both SCF and SSF behaviour can be described as above:

- Independently from the BCSM processing, the SCF tells the SSF with the RequestReportUTSI operation to report to it all the USI Information with a given *ServiceIndicator* value.

- Then, once it receives an USI, the CCF/SSF compares the *ServiceIndicator* value received with the previously indicated *ServiceIndicator* value. If they coincide, then the SSF reports the USI to the SCF with the ReportUTSI operation.

In this case, the required *ServiceIndicator* value is explicitly indicated by the SCF and stored at the SSF level; this data is in a table associated with the OCCRUI FSM.

> NOTE: The STUI/UTSI is conveyed in the appropriate signalling message depending on the phase of the call on the dedicated leg (e.g. on the BCSM processing).

6.7.1.2 Second case: the USI information is an additional and optional information

The USI information does not impact the usual DP processing. The USI is only an optional parameter that the SSF introduces in the following information flows after analysing the corresponding *ServiceIndicator* value to make sure that the received USI is targeted to this particular Service Logic.

- InitialDP.

The SSF checks if the *ServiceIndicator* value of the USI coincide with the *ServiceIndicator* value contained within the Service related data table (trigger table).

NOTE: The STUI/UTSI is conveyed in the appropriate signalling message depending on the phase of the call on the dedicated leg (e.g. on the BCSM processing).

6.7.1.3 Synthesis

Table 18 and the SDL diagram beyond summarize the two cases.

Table 18

Scenario	OCCRUI FSM	Service Indicator indicated by the User	Service Indicator value of reference	IF sent to the SCF
USI dialogue	Monitoring USI	Indicated in the STUI *Dynamic*	Stored in the data table associated with the FSM for USI	ReportUTSI
USI = additional information	Idle	Predefined	Explicitly indicated in the IN related data table of the IN service *Static in the SSF*	*InitialDP*
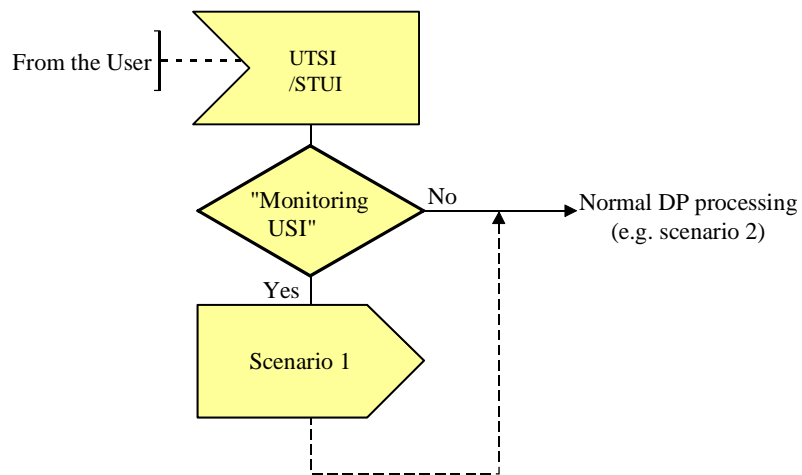
Figure 35

7 SCF Model
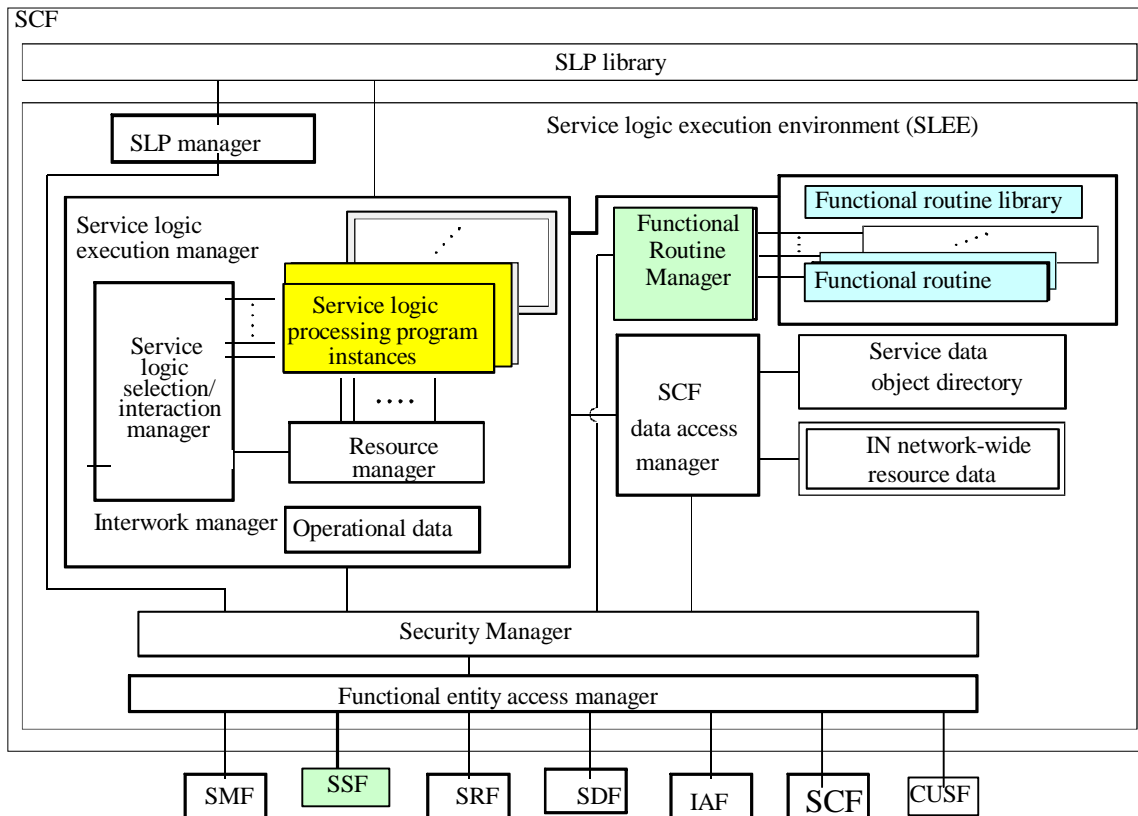
Figure 36: SCF Model

7.1 Introduction

The purpose of the SCF model shown in figure 36 is to provide a framework for the understanding of service logic processing with respect to the SCF. The SCF model shows a conceptual model of the SCF and is not intended to imply an actual implementation of the SCF.

The prime function of the Service Control Function (SCF) is the execution of service logic, which is provided in the form of Service Logic Processing programs (SLPs).

The SCF platform provides a Service Logic Execution Environment (SLEE) on which a SLPs runs to provide service processing.
For each of the components shown in figure 36 a detailed description is provided in EN 301 931-1.

7.2 SCF Model Components

For each of the components shown in figure 36 a detailed description is provided in EN 301 931-1.

Here only additional information is provided as stated below regarding functional routines that may be invoked by SLP Instances.

7.2.1   Service Logic Processing program Instance (SLPI)

When an SLP is selected and invoked, it is referred to as a Service Logic Processing program Instance (SLPI). In contrast to an SLP, a corresponding SLPI is a dynamic entity that actively controls the flow of service execution and invokes SCF functional routines.

Functional routines are programs in the SCF that can be invoked by SLPIs to cause a sequence of actions to be performed in the network in support of service execution. This sequence of actions provides the functionality defined for a service independent building block (SIB) on the Global Functional Plane. Therefore, functional routines are considered to be service independent. Potential categories of functional routines are described in the following clause entitled «Functional Routine Categories ».

7.2.2   Functional Routine Categories

The following categories of functional routines are a framework for describing the SCF functions accessible to SLPIs.

7.2.2.1   SLPI management functional routines

- Functional routines to facilitate SLPI initialization and termination;

- functional routines to invoke other SLPs.

7.2.2.2   SLPI communication functional routines

- Functional routines to support communication between SLPIs.

7.2.2.3   Timer management functional routines

- Functional routines to retrieve the current time and date;

- functional routines to manage asynchronous timers in the SCF;

- functional routines to block the invocation of an SLP for a certain defined period.

7.2.2.4   Data management interface functional routines

- Functional routines to access and manipulate SCF data (i.e. service data object directory and IN network-wide resource data) and network data (i.e. in an SDF) globally and uniformly via the SCF data access manager.

7.2.2.5   Asynchronous event handling functional routines

- Functional routines to perform appropriate functions in response to asynchronous events (e.g. events reported by other functional entities, SLPI execution error events, and internal SCF events);

- functional routines to facilitate termination of a service execution and initialization of related resources.

7.2.2.6   Connection management functional routines

- Functional routines to manipulate legs and connection points via interaction with the IN-feature manager in the SSF.

7.2.2.7   Specialized resource management functional routines

- Functional routines to access and use specialized network resources globally and uniformly via the SLEM resource manager (interacting with the SRF).

7.2.2.8   OAM functional routines

- Functional routines to respond to request for Operation, Administration & Maintenance (OAM) activities and gather OAM-related information (e.g. data collection, traffic management, error handling, charging) through interactions with the SMF.

8   FSM for SSF
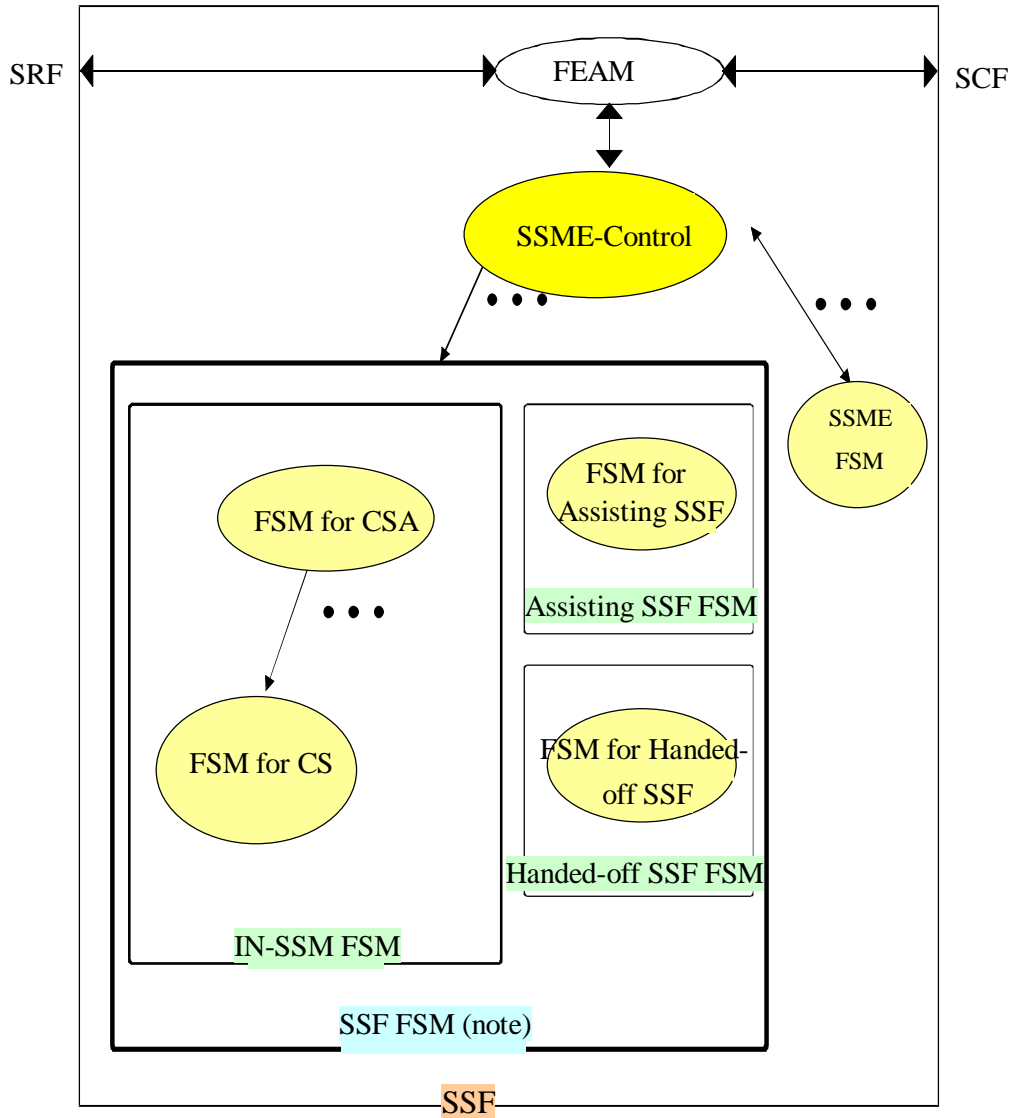
CSA - Call Segment Association

CS   - Call Segment

NOTE: One of the three possible FSMs (either IN SSM, Assisting SSF or Handed-off SSF) is selected.

Figure 37: SSF FSM structure

Functional Entity Access Manager (FEAM):
For a description of the FEAM refer to EN 301 931-1.

SSF System Management Control (SSME-Control):
The SSME-Control handles as responder the ActivityTest initiated from the SCF to the SSF.
The ActivityTest operation applies to call context transactions only.

SSF Finite State Machines (SSF FSMs):
In case interpretations for the FSM descriptions in the following differ from detailed operation procedures and the rules for using of TCAP service, the statements and rules contained in the "Operation Procedures" in clause 11 and the "Services assumed from TCAP" in clause 15 shall be followed.

Call related functions:

The SSME-control interfaces to the various inter SSF FSMs.

As shown in figure 37, an instance of an SSF FSM is either:

  a) an IN Switching State Model (IN SSM) FSM;

  b) an Assisting SSF FSM;

  c) an Handed-off SSF FSM.

The SSF FSM passes call handling instructions to the related instances of the BCSM as needed. DPs may be dynamically armed as Event DPs, requiring the SSF FSM to remain active. At some point, further interaction with the SCF is not needed, and the SSF FSM may be terminated while the BCSM continues to handle the call as needed. A later TDP in the BCSM may result in a new instance of the SSF FSM for the same call.

Consistent with the single-ended control characteristic of IN service features, the SSF FSM only applies to a functionally separate call portion (e.g. the IN service instance is triggered in the originating BCSM or the terminating BCSM in a two-party call, but not both.

The following clauses only addresses call associated relationships as supported by a IN CS-3 connection control IN-SSM.

  a) An CCF/SSF can have call associated relationships with multiple SCFs, and an SCF can have call associated relationships with multiple CCF/SSFs.
  Each relationship is treated as a one-to-one relationship.

  b) When the CCF/SSF initiates a relationship, it reports the state of the BCSM in which the TDP was detected. The state information that is included in the operations between the CCF/SSF and the SCF is defined by the parameters included in the operations, based on the analysis of IN requirements and detailed DFP modelling.

  c) Once a relationship is established between the CCF/SSF and the SCF, the SCF can request the CCF/SSF to monitor for and report subsequent events (i.e. arm EDPs), as well as to stop monitoring (i.e. disarm EDPs).

SSF Management Finite State Machines (SSME FSMs):
The management functions related to the execution of operations received from the SCF are executed by the SSF Management Entity (SSME). The SSME comprises a SSME-Control and several instances of SSME FSMs.

The different contexts of the SSME FSMs may be distinguished based on the address information provided in the initiating operations. In the case of service filtering this address information is given by filteringCriteria, i.e. all ActivateServiceFiltering operations using the same address, address the same SSME-FSM handling this specific service filtering instance. For example ActivateServiceFiltering operations providing different filtering Criteria cause the invocation of new SSME-FSMs by the SSME-Control.

The SSME FSMs and SSF FSMs are described in the following clauses.

8.1   SSF management finite state model (SSME FSM)

The SSME FSM state diagram is described in figure 38:
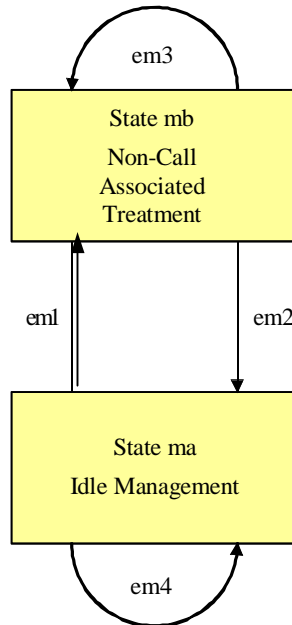
Figure 38: SSME FSM state diagram

The **SSME FSM** is independent of the individual **FSMs for CS**.

In the Idle Management state the following operations may be received from the SCF and processed by the SSME FSM with no resulting transition to a different state (transition em4):

- **RequestCurrentStatusReport;**

- **ManageTriggerData;**

- **CreateOrRemoveTriggerData.**

The **ManageTriggerData** and **CreateOrRemoveTriggerData** operations can only be received outside a call context transaction.

The Non-Call Associated Treatment state is entered from the Idle Management state when one of the following non-call associated operations is received (transition em1):

- **RequestEveryStatusChangeReport;**

- **RequestFirstStatusMatchReport;**

- **ActivateServiceFiltering;**

- **CallGap.**

The **CallGap** operation can be received inside as well as outside a call context transaction. The **ActivateServiceFiltering** operation can be received outside a call context transaction only.

During this state the following events can occur:

- given that service filtering is active, the SSF shall send a service filtering response to the SCF: the SSME FSM remains in this state (transition em3);

- given that service filtering is active, the SSF shall increment a counter; the SSME FSM remains in this state (transition em3);

- given that service filtering is active and the service filtering duration expires: the SSME shall send a **ServiceFilteringResponse** operation to the SCF; the SSME FSM moves to the Idle Management state (transition em2);

- given that status report is active, as previously requested by a **RequestEveryStatusChangeReport** operation, the SSF sends a **StatusReport** operation with reportCondition (optional) when the resource status changes; the SSME FSM remains in this state (transition em3);

- given that status report is active, as previously requested by a **RequestFirstStatusMatchReport** operation, the SSF sends a **StatusReport** operation with reportCondition (optional) when the resource status matches; the SSME FSM transits to Idle Management state (transition em2);

- given that status report is active, as previously requested by a **RequestFirstStatusMatchReport** or a **RequestEveryStatusChangeReport** operation, the SSF sends a **StatusReport** operation with reportCondition set to "cancelled" when the monitor is cancelled by receiving a **CancelStatusReportRequest** operation from the SCF; the SSME FSM moves to the Idle Management state (transition em2);

- given that status report is active, as previously requested by a **RequestFirstStatusMatchReport** or a **RequestEveryStatusChangeReport** operation, the SSF sends a **StatusRepor**t operation with reportCondition set to "timerExpired" when the monitor duration expires; the SSME FSM moves to the Idle Management state (transition em2);

- if call gap related duration timer expires, the SSME FSM moves to the Idle Management state (transition em2);

- given that call gap/service filtering is active, another **CallGap/ActivateServiceFiltering** operation having the same gapping/filtering criteria can be received by the SSF: the second "filter" or "gap" replaces the first one (transition em3) unless the duration timer value is equal to zero, in which case the SSME FSM moves to the Idle Management state (transition em2).

All other operations have no effect on the SSME-FSMs; the operations are passed by the SSME-Control to the relevant FSM for CSA.

8.2   IN SSF switching state model (IN -SSM) FSM

The IN-SSM FSM consists of a FSM for a Call Segment Association (FSM for CSA). The FSM for CSA creates one or more sub FSMs for Call Segment (FSM for CS).

NOTE 1: Within this clause the term *CPH operations* is used.

CPH operations are the following:

- DisconnectLeg;

- SplitLeg;

- MoveLeg;

- MergeCallSegments;

- MoveCallSegments.

General rules:
General rules and procedure principles for inclusion of Call Party Handling (CPH) capabilities into the FSM for CS are addressed here.

- **Timer treatment:**

The use of a Timer to guard the SSF-SCF association (TC dialogue) or to prevent against excessive call suspension shall be done at the CS level.

- **Change of Connection View:**

  - The change of a Connection View (CV) may imply a change of CSCV state and is initiated either from the End User side (e.g. midcall DP) or from the SCF (SCF initiated CV change).

  - The SCF may change the CV by sending of one of the following operations:

    - **Connect;**

    - **ContinueWithArgument;**

    - **DisconnectLeg;**

    - **InitiateCallAttempt;**

    - **SplitLeg;**

    - **MergeCallSegments;**

    - **MoveCallSegments;**

    - **MoveLeg;**

    - **ReleaseCall;**

    - **SelectFacility.**

- **SCF and Connection View (CV) state changes:**

  The SCF is informed about changes of the CV via:

  - EventReportBCSM, EntityReleased, CallInformationReport.

  - CPH operations when ReturnResults are sent by the SSF on a successful change of the CV.

- **SCF leg control:**

  - The SCF may control a leg for which at least the O/T:Disconnect DP or O/T_Abandon DP is armed or one report (CallInformationReport/ApplyChargingReport) is pending. (Lifetime supervision of the leg).

  - The SCF shall have a connection view of the legs involved in the call.
    This is done by informing the SCF about the leg state changes e.g. the disconnection of a leg. It shall not be allowed to have legs at a connection point which are not visible to the SCF (i.e. no DP armed/report pending).

  - The SCF becomes aware of each CSCV state transition in the SSF due to call handling events by the report of armed DPs on a per leg basis. If the SLPI in the SCF does not arm all DPs it cannot have a correct view of the evolution of its corresponding connection view in the SSF. In order to secure a correct connection view for service execution the SCF is to be aware of the present CSCV state in the SSF via DP arming and DP reporting.

- **Resumption Counter Rules:**

  The number of operations sent by the SCF to resume call processing (i.e. when the FSM for CS is in any Waiting For Instructions state) shall be equal to the number of events that caused the suspension of the call process.

  Events that cause the suspension of the call processing are signalling events armed as TDP-Rs or Event Detection Point - Requests (EDP-Rs), or the processing of a CPH operation sent by the SCF.

  The number of required resumptions will be incremented by 1 in case of a TDP-R or EDP-R. In case of a CPH operation the number of required resumptions will be set to one if it is still 0. Otherwise the number of resumptions remains unchanged.

  In addition The SSF FSM stores information about the events that require resumption(DP, leg; CPH) and keep track of the order of events to be able to correlate the received resumptions with the events. In case of an CPH event the event information of the event that has to be resumed next will be overwritten by the CPH event.

  For each received resumption (CWA, CUE) the number of required resumption will be decremented by 1 if it was a valid resumption for the event that has to be handled first. For DP events only CWA with the correct legID or CUE are valid. For CPH resumption only CWA with CSID or CUE are valid. In case an invalid resumption is received by the SSF an error has to be sent to the SCF.

  If the last leg exits from the CallSegment the resumption of a 'release' DP is performed independent of the order of events. That means a resumption for a Disconnect DP is accepted also if there was a CPH event stored before.

  The processing of a Connect, CollectInformation or SelectFacility causes the number of resumptions required to be set to 0 and the call processing to be resumed. All stored resumption events are discarded.

- **CPH procedure principles for the FSM for CS:**

  - The import of a leg (including EDPs or pending reports) in a CS (target CS) shall not affect the events armed (including EDPs or pending reports) for other leg(s) residing in the same CS.

  - The **FSM for CS** (one FSM per Call Segment and Connection Point) shall not know how many legs are connected to the CS. The **FSM for CS** exists as long as at least one report is pending or a DP is armed.

- **Procedure principles for User Interaction:**

  - In the states Waiting for End of User Interaction(WFI/MON) and Waiting for End of Temporary Connection(WFI/MON) where a SRF resource is connected to the CS, it is not allowed to change the Connection View state e.g. via CPH, Continue, ContinueWithArgument or Connect operations.
    The following CPH operations are not allowed:

    - DisconnectLeg;

    - SplitLeg;

    - MoveLeg;

    - MergeCallSegments;

    - MoveCallSegments.

  - In the states Waiting for End of User Interaction (WFI) and Waiting for End of Temporary Connection(WFI) where a SRF resource is connected to the CS and call processing is suspended, the only call processing operations allowed are:

    ContinueWithArgument;
    Continue

    Per one CS only one connection is allowed to a resource (i.e. User Interaction or Temporary Connection). It implies that only one SRF resource can be connected to a CS at one time. The resource can either be connected to one leg or to the connection point in the CS.

NOTE 2:   PromptAndCollectUserInformation and PromptAndReceiveMessage operations are restricted to be applied for a CS with only one joined leg.

- **Procedure principles for CPH operations:**

  - All CPH operations received by the FSM for CS shall cause a transition to the state Waiting For Instructions. Exception: This does not apply to the MoveCallSegments operation since this operation is received by the **FSM for CSA** and not the **FSM for Call Segment**.

  - If one of the following operations DisconnectLeg, SplitLeg, MoveLeg or MergeCallSegments is received in the Monitoring state the FSM for CS shall process the received operation and then it shall transit to the Waiting for Instructions state (valid for SCF initiated changes of Connection view states).

  - The receipt of one of the above operations in the state Waiting for Instructions shall not cause the change of the state Waiting for Instructions. Therefore all operation sequences shall be finalized by an operation which changes the state into Monitoring (e.g. ContinueWithArgument, Connect), except the case that a operation causes the transition to state Idle. (e.g. DisconnectLeg for the last leg).

  - When the SplitLeg operation is received in the Idle state for the target FSM for CS, the FSM for CS shall process the received operation and shall then transit to the Waiting for Instructions state.

The following principles shall apply to timed call release:

- User interaction e.g. tone/announcement sending prior to call release is possible. The "time count down" (time duration to release) in the SSF for the delayed call release is started when the ReleaseCall operation is received and accepted.
  The SCF initiated timed call release is performed according to the requested 'time duration to release'. See note 3. In order to secure a call release right on the requested time, a possible preceding user interaction (e.g. play announcement) shall be terminated before the call release is to be executed, or a 'forced release' should be requested.

NOTE 3:   The call may be released before this timer expires in case e.g. the user or another service involved in the call makes a request for a release of the call.

Each FSM and the according states are discussed in the following clauses.

General rules applicable to more than one FSM state are addressed here.

One or a sequence of components received in one or more TCAP messages may include a single operation or multiple operations, and is processed as follows:

- Process the operations in the order in which they are received.

- Each operation causes a state transition independent of whether or not a single operation or multiple operations are received in a message.

- **Operation examination and buffer handling:**
  The SSF examines subsequent operations in the sequence. As long as sequential execution of these operations leaves the FSM in the same state, it shall execute them (e.g. RequestReportBCSMEvent). If a subsequent operation causes a transition out of the state then the following operations shall be buffered until the current operation has been executed. In all other cases, await an event that causes a transition out of the current state (such an event is the completion of operation being executed, or the reception of an external event). An example of this is as follows:

EXAMPLE: The SSF receives the operations FurnishChargingInformation, ConnectToResource, and PlayAnnouncement in a component sequence inside a single TC message. Upon receipt of this message, these operations are executed up to and including ConnectToResource while the SSF is in the Waiting For Instructions state. As the ConnectToResource operation is executed (and when, or after the FurnishChargingInformation operation has been completed), the FSM for CS will transit to the Waiting For End Of User Interaction state. The PlayAnnouncement operation is relayed to the SRF while the SSF is in Waiting For End Of User Interaction state.

- **Error handling:**
  If there is an error in processing one of the operations in the sequence, the **FSM for CS** processes the error (see below) and discards all remaining operations in the sequence. This discarding of operations is made independent of if they are addressed for the same of different CSs within the CSA.

- If an operation is not understood or is out of context (i.e. violates the SACF rules defined by the **FSM for CS**) as described above, ABORT the interaction by sending of a TCAP ABORT at the CSA level or an operation EntityReleased at the CS level. For example when the FSM for CS applies to an originating BCSM then receiving SelectFacility operation would be out of context since this applies only to the terminating half of the BCSM.

In any state, if there is an error in a received operation, the maintenance functions are informed and the **FSM for CS** remains in the same state as when it received the erroneous operation; depending on the class of the operation, the error could be reported by the SSF to the SCF using the appropriate component (ITU-T Recommendation Q.774).

8.2.1   Finite State Model for Call Segment Association (CSA)

Figure 39 shows the state diagram of the **FSM for CSA** in the SSF part of the SSP during the processing of an IN call/attempt.
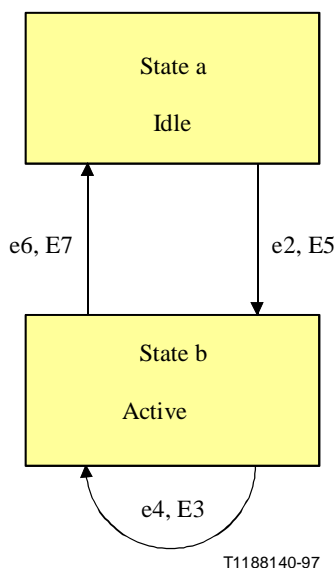
Figure 39: FSM for Call Segment Association

An instance of the FSM for CSA is created by SSME-Control when:

- an indication of a new call attempt is received from a user;

- a message related to a new transaction containing an InitiateCallAttempt or CreateCallSegmentAssociation operation is received from the SCF.

The FSM for CSA state diagram contains the following transitions (events):

e2	TDP-R encountered	

E3 any operation received from SCF except transit to state Idle (includes InitiateCallAttempt)

e4 EDP-R encountered

    EDP-N last encountered except last one from last CS

    any response and report

E5 InitiateCallAttempt (received in state Idle)

    CreateCallSegmentAssociation received

e6 last EDP-N from last CS and no other reports pending

E7 any operation received from SCF which causes no remaining CS (e.g. ReleaseCall, MoveCallSegments or DisconnectLeg)

The CSA state diagram contains the following states:

   State a   Idle

   State b   Active

Relationship to CSA CV:
The CSA also encompasses the CSA Connection View transition diagram model (CSACV).
The states identified for the CSA CV transition diagram are: "Null", "One Segment" and "Multi Segments".
The state "Null" represents the condition where no call processing is active; and is associated with the FSM for CSA state "Idle".
The states "One Segment" and "Multi-Segments" represents a call segment association containing one respective multiple call segment under SLPI control in a relationship between SCF and SSF; and is associated with the FSM for CSA state "Active".

8.2.1.1 State a: Idle

The **FSM for CSA** enters the Idle state under a variety of conditions, as described below.

The **FSM for CSA** enters the Idle state when sending or receiving an ABORT TCAP primitive due to abnormal conditions in the state Active.

The **FSM for CSA** enters the Idle state when one of the following occurs:

- when all the **FSM for Call Segment** instances associated with the **FSM for CSA** instance are released.

During this state the following call-associated events can occur:

- The following operations may be received from the **FSM for CS**, causing a state transition to the Active state (transition e2):

    - an **InitialDP** is received indicating an encountered TDP-R. The received operation is sent to the SCF.

    The rules for DP processing are described in clause "DP Processing".

- A message related to a new transaction containing an **InitiateCallAttempt** or a **CreateCallSegmentAssociation** operation is received from the SCF: in this case the **FSM for CSA** moves to the state Active (transition E5).

Any other operation received from the SCF while the **FSM for CSA** is in Idle state, i.e. while the **FSM for CS** instance does not exist, should be treated as an error. The event should be reported to the maintenance functions and the transaction should be aborted according to the procedure specified in TC.

8.2.1.2   State b: Active

This state is entered from the Idle state by detecting a TDP-R (transition e2), from the Idle state on receipt at the SSF of a TC_Begin indication primitive containing an **InitiateCallAttempt** or a **CreateCallSegmentAssociation** operation from the SCF (transition E5).

As an operators option, the SSF may start an Activity Test timer (Tati) for supervising the activity of the dialogue established with the SCF. This timer is restarted each time an operation is received from the SCF and is stopped when the CSA instance is deleted.

In this state the **FSM for CSA** handles instructions from the SCF and events which are received from the **FSMs for CS**.

During this state the following events can occur:

- The receipt of an END or ABORT primitive from TCAP has no effect on the call; the call may continue or be completed with the information available. In this case, the **FSM for CSA** transits to the Idle state (transition E7), disassociating the **FSM for CSA** from the call, except for the case where a 'timed disconnect' has been requested in the ReleaseCall operation. In the latter case the transition to Idle and the disassociation of the FSM for CSA is postponed until the timer (Tdisconnect) in the CSA expires (unless call release occurs beforehand) and the requested time controlled call release is executed.

- An operation is received from the SCF: The **FSM for CSA** acts according to the operation received as described below.

- An operation is received from the **FSM for CS**: The **FSM for CSA** acts according to the operation received as described below.

- Expiry of the Activity Test timer Tati, (if set) will cause the sending of the ActivityTest operation, the **FSM for CSA** will remain in the same state.

- Expiry of the ActivityTest operation timer, this means that the relationship with the SCF for the dialogue no longer exists. In this case, the **FSM for CSA** transits to the Idle state (transition e6) disassociating the FSM for CSA from the call. The subsequent call treatment depends on the information in the trigger description, e.g. routing the call to a termination announcement.

The following operations may be received from the SCF and processed by the SSF with no resulting transition to the idle state (transition E3):

- **ActivityTest;**
- **ApplyCharging;**
- **CallInformationRequest;**
- **Cancel(invokeID and optional callSegmentToCancel);**
- **ConnectToResource;**
- **DisconnectForwardConnection;**
- **DisconnectForwardConnectionWithArgument;**
- **EstablishTemporaryConnection;**
- **FurnishChargingInformation.**

NOTE 1: It is network operator specific whether the operation is forwarded to any **FSM for CS** instance using the internal coding of the OCTET STRING or not.

- **InitiateCallAttempt.**

NOTE 2: In this case the **FSM for CSA** creates a new **FSM for CS** instance and transmits the event to it.

- **MergeCallSegments.**

NOTE 3:　In this case, the SSF deletes the "source" Call Segment and connects the Leg in the "source" Call segment with the "target" Call Segment. The **FSM for CSA** transmits the event to the FSM instance for the "source" CS and releases the FSM instance. Furthermore, the **FSM for CSA** transmits the event to the FSM instance for the "target" CS.

- **MoveCallSegments (for target CS);**

- **MoveLeg.**

NOTE 4:　In this case, the SSF moves the leg from the 'source' Call Segment to the 'target' Call Segment with which the source Call Segment is associated. The **FSM for CSA** transmits the event to the 'source' **FSM for CS** instance and to the 'target' **FSM for CS** instance.

- **PlayAnnouncement;**

- **PromptAndCollectUserInformation;**

- **PromptAndReceiveMessage;**

- **RequestNotificationChargingEvent;**

- **ResetTimer;**

- **ScriptClose;**

- **ScriptInformation;**

- **ScriptRun;**

- **SendChargingInformation;**

- **SetServiceProfile;**

- **SplitLeg.**

NOTE 5:　In this case, the SSF creates a Call Segment and connects the split Leg with the Call Segment. The **FSM for CSA** transmits the event to the FSM instance for the "source" CS. Furthermore, the **FSM for CSA** creates a new FSM instance for the 'target' Call Segment and transmits the event to the FSM.

The following operations may be received from the SCF, causing a state transition either to the same state if an **FSM for CS** instance exists after this event was processed in the FSM (transition E3), or to the Idle state if all the **FSM for CS** instances associated with the **FSM for CSA** instance transit to the Idle state (transition E7).

- **Cancel(allRequests);**

- **CollectInformation;**

- **Connect;**

- **Continue;**

- **ContinueWithArgument;**

- **DisconnectLeg;**

- **MoveCallSegments (for source CS);**

- **RequestReportBCSMEvent;**

- **SelectFacility.**

The **ReleaseCall** operation (immediate request) may be received from the SCF. For the case the whole CSA is to be released, the **FSM for CSA** shall instruct all the relevant **FSM for CS** instances to clear the call and ensure that any CCF resources allocated to the call have been de-allocated, then continue processing as follows:

- if the last CS has been released, and if neither CallInformationReport nor ApplyChargingReport operation has been requested, the **FSM for CSA** transits to the Idle state (transition E7);

- if the last CS has been released, and if CallInformationReport or ApplyChargingReport operation has been requested, the SSF sends each operation which has been requested from SCF, and then the **FSM for CSA** transits to the Idle state (transition E7).

Timed disconnect:

If a 'timed disconnect' is requested in ReleaseCall, an application timer 'Tdisconnect' is started on CSA level upon receipt of this operation. Any new operation received from the SCF after the ReleaseCall operation is considered as an error. The execution of the ReleaseCall is postponed until the timer 'Tdisconnect' expires. When the timer Tdisconnect expires the ReleaseCall is handled (immediate request) as described above.

When invoking the ReleaseCall operation with a time-to-release parameter, the service logic may decide (but does not need to) to close the dialogue (using a TC-End-Req primitive) in order to be disengaged from the call. In such a case no pending reports (ApplyChargingReport/CallInformationReport) will be received by the service logic. The timed disconnect only applies if the ReleaseCall operation applies to all call segments in the CSA. In case the dialogue is closed (using a TC-End-Req primitive) and the application timer 'Tdisconnect' is started on CSA level, then the CSA and associated CSs are not disassociated from the call because the execution of the ReleaseCall is to be postponed until the timer 'Tdisconnect' expires.

CSs are created or deleted by the CSA, if necessary, then the operations are passed to the appropriate CS and are processed there.

The following operations may be received from one of the CS, causing a state transition either to the same state if an **FSM for CS** instance exists after this event was processed in the FSM (transition e4), or to the Idle state if all the **FSM for CS** instances associated with the **FSM for CSA** instance transit to the Idle state (transition e6). The operations are sent to the SCF:

- **ApplyChargingReport;**

- **CallInformationReport;**

- **EntityReleased;**

- **EventReportBCSM.**

The following operations may be received from one of the CS with no resulting transition to the idle state (transition e4), the operations are sent to the SCF:

- **EventNotificationCharging;**

- **ReturnResult for PromptAndCollectUserInformation;**

- **ReturnResult for PromptAndReceiveMessage;**

- **ScriptEvent;**

- **SpecializedResourceReport.**

Any other operation received in this state should be processed in accordance with the general rules in 8.2.

8.2.2   Finite State Model for Call Segment
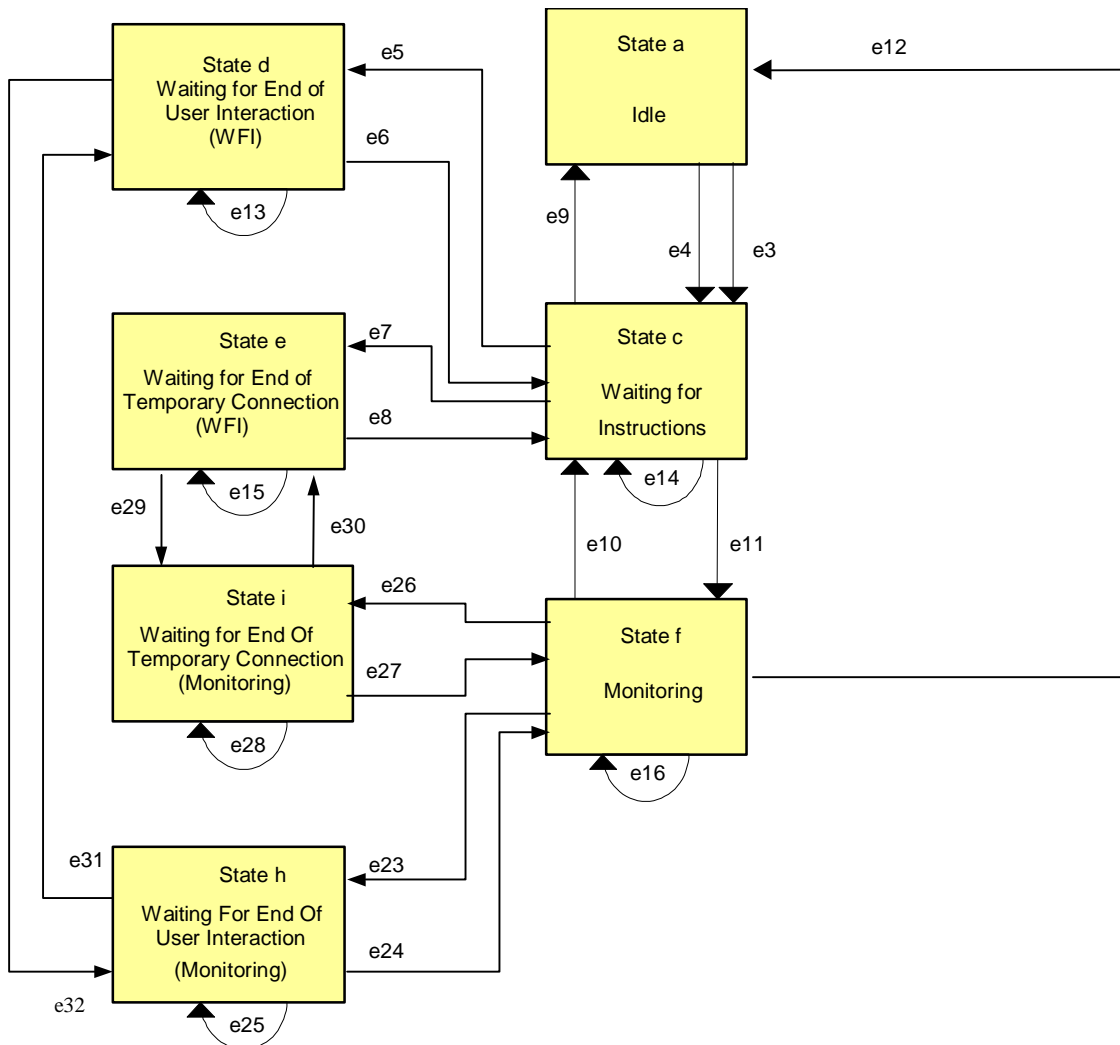
NOTE: Only the main transitions are shown. A transition to Idle is possible from any state. More detailed
         information concerning the possible state transitions can be found in the clauses describing the FSM for
         CS states.

Figure 40: FSM for Call Segment

The SSF state diagram for the CS contains the following transitions (events):

e3   InitiateCallAttempt received, SplitLeg received (transition when "target" CS)
e4   TDP-R encountered
e5   User interaction requested
e6   User interaction ended
e7   Temporary connection created
e8   Temporary connection ended
e9   Idle return from Waiting for Instructions
e10   EDP-R encountered
e11   Routing instruction received
e12   EDP-N last (see note 1) encountered or disarmed or ReleaseCall received or Cancel(allRequests)
            received
e13   Waiting For End Of User Interaction state no change
e14   Waiting For Instructions state no change
e15   Waiting For End Of Temporary Connection state no change
e16   Monitoring state no change
e23   User interaction requested

e24	User interaction ended
e25	Waiting For End Of User Interaction state no change
e26	Temporary connection created
e27	Temporary connection ended
e28	Waiting For Temporary Connection state no change
e29	Continue has been issued
e30	EDP-R encountered
e31	EDP-R encountered
e32	Continue has been issued

NOTE 1: The "last EDP-N" means that there are no other EDPs which may be encountered when an EDP-N was detected. Some of the EDPs are automatically disarmed if another EDP is encountered. The EDPs which are automatically disarmed depend on which EDP is encountered. An example is the case of the EDPs O_Answer, O_No_Answer, RouteSelectFailure or O_Called_Party_Busy. If any of these EDPs are encountered, all the other EDPs of this list are automatically disarmed.

NOTE 2: The **FSM for CS** may be in any state, except Idle if the O/T_Abandon DP, O_Term_Seized DP, O/T_Answer DP, O/T_Disconnect DP -or O/T_MidCall DP (when requested as "inAnyState") is armed and encountered. This ensures that the SCF will receive the events in the correct order. For example in case the **FSM for CS** is in "Waiting for Instructions" state due to an O-MidCall EDP-R (SendCall) being reported and in this state O-Term seized and O-Answer events are detected followed by the O_Disconnect event.

NOTE 3: Non-Call Associated Treatment events can be encountered in any state, the **FSM for CS** remains in the same state.

The SSF state diagram contains the following states:

State a	Idle
State c	Waiting For Instructions
State d	Waiting For End Of User Interaction (WFI)
State e	Waiting For End Of Temporary Connection (WFI)
State f	Monitoring
State h	Waiting For End Of User Interaction (Monitoring)
State i	Waiting For End Of Temporary Connection (Monitoring)

Calling Party Abandon:
In any state (except Idle), if the calling party abandons the call before it is answered (i.e. before the Active PIC in the BCSM), then the **FSM for CS** instance shall instruct the CCF to clear the call and ensure that any CCF resources allocated to the call are de-allocated, processing shall continue as shown in table 19.

Table 19

Abandon DP	operation sent to SCF	state transition
not armed	**[CallInformationReport] +** **[ApplyChargingReport]**	to Idle note 2
armed as EDP-R	**[CallInformationReport] +** **[ApplyChargingReport] +** **EventReportBCSM**	to WaitForInstructions
armed as EDP-N	**EventReportBCSM +** **[CallInformationReport] + [ApplyChargingReport]**	to Idle note 2
NOTE 1: The operations in brackets "[]" are only sent if the reports are pending.		
NOTE 2: The transition to Idle applies for the case that no more DPs are armed and there are no pending reports for the call segment.		

Call party Disconnect:

In any state (except Idle), if a call party disconnects from a stable call (i.e. from the Active PIC in the BCSM), then the FSM for CS **shall process this event as shown in table 20.**

Table 20

Disconnect DP	operation sent to SCF	state transition
not armed for that specific leg	[**CallInformationReport**] + [**ApplyChargingReport**]	to Idle note 1
armed as EDP-R for that specific leg	[**CallInformationReport**] + [**ApplyChargingReport**] + **EventReportBCSM**	to WaitForInstructions
armed as EDP-N for that specific leg	**EventReportBCSM** + [**CallInformationReport**] + [**ApplyChargingReport**]	to Idle note 1
NOTE 1: The operations in brackets"[]" are only sent if the reports are pending. NOTE 2: The transition to Idle implies that the CS instance is deleted as no more DPs are armed and there are no pending reports for the call segment.		

In any state (except Idle), an **EventNotificationCharging** can be sent to the SCF, if previously requested by a RequestNotificationChargingEvent and if the charging event has been detected by the CCF. In this case no state transition takes place.

Tssf Timer:
Each **FSM for CS** instance has an application timer, T_{SSF}, whose purpose is to prevent excessive call suspension time and to guard the association between the SSF and the SCF.

If necessary, the Timer T_{SSF} may be set in the following cases:

- When the SSF sends an InitialDP operation for a TDP-R State c: Waiting For Instructions. While waiting for the first response from the SCF, the timer T_{SSF} can be restart only once by a ResetTimer operation. Subsequent to the first response, the timer can be restart any number of times.

- When the **FSM for CS** enters the Waiting For Instructions state under any other condition then the ones listed in the previous case. In this case the SCF may restart the T_{SSF} timer using the ResetTimer operation any number of times.

- When the SSF enters the Waiting For End Of User Interaction state or the Waiting For End Of Temporary Connection state. In these cases the SCF may restart T_{SSF} using the ResetTimer operation any number of times.(OPTIONAL).

   NOTE 4: This "OPTIONAL" means that the application timer T_{SSF} is optionally set. Whether it is used or not depends on implementation. But it must be synchronized with $T_{SCF-SSF}$ in the SCSM.

In each of the above cases, T_{SSF} may have different values as defined by the application.

When receiving or sending any operation which is different from the above, the **FSM for CS** instance shall restart T_{SSF} with the last used value. This value is either one associated to the different cases as listed above, or received in a ResetTimer operation, whatever occurred last. In the state Monitoring T_{SSF} is not used.

On expiration of T_{SSF} the **FSM for CS** transits to the Idle state and the CCF progresses the BCSM if possible. If the **FSM for CS** was the last in the CSA the interaction with the SCF is aborted, otherwise the operation **EntityReleased** is sent to the SCF.

The valid state transitions for the **FSM for CS** are contained in the following "FSM for CS transition table".
The columns present the current state in which the operation or event may occur.
The existence of an entry indicates that in the current state the operation/event is valid.
The text of the entry indicates the transition to the **FSM for CS** state after processing the operation/event.

   NOTE 5: The table does not cover error handling (e.g. Disconnect, Abandon) and does not show how the SSF examines subsequent operations in a sequence and execute and buffers operations in accordance with the general rules described in clause 8.2.

Table 21: "FSM for CS transition table" for valid FSM for CS transitions

state: ⇒ Operations: ⇓	State a Idle	State c WfI	State d WfEoUI (WFI)	State e WfEoTC (WFI)	State h WfEoUI (MON)	State i WfEoTC (MON)	State f Mon
ApplyCharging		same	same	same	same	same	same
CallInformationRequest		same					
Cancel(allRequests)		same					Idle
CollectInformation		Mon (note 3)					
Connect		Idle (note 2), Mon (note 3)					
ConnectToResource		WfEoUI (WFI)					WfEoUI (MON)
Continue		Idle (notes 1, 2), Mon (notes 1, 3)	Idle (notes 1, 2, 10, 13) WfEoUI (MON) (notes 1, 3, 10)	Idle (notes 1, 2, 10, 13) WfEoTC (MON) (notes 1, 3, 10)			
ContinueWithArgument		Idle (note 2), Mon (note 3)	Idle (notes 2, 13) WfEoUI (MON) (notes 3, 10)	Idle (notes 2, 13 WfEoTC (MON) (notes 3, 10)			
DFCWithArgument			WfI	WfI	Mon	Mon	
DisconnectForwardConnection			WfI (note 9)	WfI (note 9)	Mon (note 9)	Mon (note 9)	
DisconnectLeg(last joined leg)		Idle					Idle
DisconnectLeg(not last joined leg)		same					WfI
EstablishTemporaryConnection		WfEoTC (WFI)					WfEoTC (MON)
FurnishCharging Information		same	same	same	same	same	same
InitiateCallAttempt	WfI						
MergeCallSegments (targetCS)		same					WfI
MergeCallSegments (sourceCS)		Idle					Idle
MoveCallSegments		same (note 14)					same or WFI (note 14)
MoveLeg (last leg for sourceCS)		Idle					Idle
MoveLeg (targetCS; not last leg for sourceCS)		same					WfI
ReleaseCall (for a CS)		Idle	Idle (note 15)	Idle (note 15)	Idle (note 15)	Idle (note 15)	Idle
RequestNotification ChargingEvent		same	same	same	same	same	same
RequestReportBCSMEvent		same					Idle (note 2), same (note 3)
ResetTimer		same	same (note 4)	same (note 4)	same (note 4)	same (note 4)	

state: ⇒ Operations: ⇓	State a Idle	State c Wfl	State d WfEoUI (WFI)	State e WfEoTC (WFI)	State h WfEoUI (MON)	State i WfEoTC (MON)	State f Mon
SelectFacility		Idle (note 2), Mon (note 3)					
SendChargingInformation		same	same	same	same	same	same
SetServiceProfile		same	same	same	same	same	same
SplitLeg (sourceCS)		same					Wfl
SplitLeg (newCS)	Wfl						
Operations for relaying to SRF ⇓							
Cancel(invokeID)			same (note 8)		same (note 8)		
PlayAnnouncement			same (note 8)		same (note 8)		
PromptAndCollect UserInformation (PaCUI)			same (note 8)				
PromptAndReceiveMessage (PaRM)			same (note 8)				
ScriptClose			same (note 8)				
ScriptInformation			same note 8)				
ScriptRun			same (note 8)				
TDP-R (InitialDP)	Wfl						
Tssf		Idle	Idle (note 4)	Idle (note 4)	Idle (note 4)	Idle (note 4)	
event - disconnect from SRF			Wfl	Wfl	Mon	Mon	
event - EDP-R		same (note 12)	same (note 12) WFI (note 11)	same (note 12) WFI (note 11)	WfEoUI (WFI) WFI (note 11)	WfEoTC (WFI) WFI (note 11)	Wfl
event - EDP-N		same (note 12)	same (note 12) WFI (note 11)	same (note 12) WFI (note 11)	Idle (note 2), same (note 3) Mon (note 11)	Idle (note 2), same (note 3) Mon (note 11)	Idle (note 2), same (note 3)
event - last pending report							Idle
event - Feature activation/hook flash (EDP-R)		same (note 7)	same (note 7)	same (note 7)	WfEoUI (WFI)	WfEoTC (WFI)	Wfl
SpecializedResourceReport			same (note 8)		same (note 8)		
ReturnResult from PaCUI			same (note 8)				
ReturnResult from PaRM			same (note 8)				

state: ⇒ Operations: ⇓	State a Idle	State c WfI	State d WfEoUI (WFI)	State e WfEoTC (WFI)	State h WfEoUI (MON)	State i WfEoTC (MON)	State f Mon
ScriptEvent			same (note 8)				

NOTE 1:   Only applicable for a single CS with no more than 2 legs, use of this operation is not valid in a multi call segment CSA.
NOTE 2:   No EDPs armed and no pending requests.
NOTE 3:   EDPs armed or pending requests.
NOTE 4:   Use of Timer Tssf in this state is optional.
NOTE 5:   Void.
NOTE 6:   Void.
NOTE 7:   Only if MidCall DP was armed to be reported in any state, except Idle.
NOTE 8:   Operations/events for relaying to SRF.
NOTE 9:   Only applicable for a single CS, use of this operation is not valid in a multi call segment CSA.
NOTE 10: Only allowed if applied user interaction operation is PlayAnnouncement. ETC or CTR in the transparent relay case is accepted in the Monitoring state, but it is up to the SCF to ensure that the UserInteraction through the assisting SRF will be limited to PlayAnnouncement.
NOTE 11: In case of DP Abandon or DP Disconnect for the leg that is connected to the SRF resource.
NOTE 12: Only if the current DP is DP O/T_Disconnect or DP O/T_Abandon (when no SRF resource is connected to the leg), O_Term_Seized DP, O/T_Answer DP.
NOTE 13: All associated SSF and SRF resources including the connection to SRF are released immediately, i.e. before call processing is resumed.
NOTE 14: For any possible change of state refer to post conditions as defined in the procedure description for the MoveCallSegments operation.
NOTE 15: Transition to "Idle" occurs immediately if *forced release* is requested, otherwise the transition occurs when the requested user interactions have been completed (i.e. ReleaseCall operation is buffered until user interaction is ended and a transition back to "Waiting for Instructions" or "Monitoring" state occurs).

The following clauses give more specific information for each state in addition to the **FSM for CS** transition table. The state transitions described below in the text gives the most important actions; for a complete specification please refer to the tables 19, 20 and 21.

8.2.2.1   State a: Idle

The **FSM for CS** enters the Idle state under a variety of conditions, as described below and in the **FSM for CS** transition table.

The **FSM for CS** enters the Idle state when the associated **FSM for CSA** instance transits to the Idle state.

The **FSM for CS** enters the Idle state when one of the following occurs:

- when the call is abandoned or one or more call parties disconnect in any other state under the conditions identified in 8.2.2;

- when an operation is processed in the Waiting For Instructions state, and no EDPs are armed and there are no outstanding report requests (transition e9). Refer to "FSM for CS transition table" for operations causing this transition.

When transiting to the Idle state, if there is a CallInformationReport and/or ApplyChargingReport pending (see State c: Waiting for Instructions), the SSF sends a CallInformationReport and/or ApplyChargingReport operation to the SCF before returning to Idle. Once in the Idle state, if status reporting is still active the SSF deactivates it, any outstanding responses to send to the SCF are discarded.

During the state Idle the following call-associated events can occur:

- indication from the CCF that an armed TDP-R is encountered related to a possible IN call/service attempt, the **FSM for CS** instance sends a generic **InitialDP** to the associate **FSM for CSA**, as determined from DP processing, and transits to the Waiting For Instructions state (transition e4).

The rules for DP processing are described in clause "DP Processing":

- an **InitiateCallAttempt** operation is received from the SCF: in this case a new **FSM for CS** instance is created by the associate **FSM for CSA**. This **FSM for CS** instance transits to the Waiting For Instructions state (transition e3);

- a **SplitLeg** operation is received from the SCF: in this case a new target **FSM for CS** instance is created by the associate **FSM for CSA**. This **FSM for CS** instance transits to the Waiting For Instructions state (transition e3).

Any other operation received from the SCF while the **FSM for CS** is in Idle state should be treated as an error. The event should be reported to the maintenance functions.

8.2.2.2   State c: Waiting For Instructions

This state is entered:

- from the Idle state, either directly as indicated above (transition e4), or on receipt of an **InitiateCallAttempt** or **SplitLeg** ('target' CS) operation from the SCF (transition e3);

- from the state Monitoring on detection of an EDP-R or one of the following CPH operations MoveLeg, SplitLeg, MergeCallSegments or DisconnectLeg (transition e10);

- from the state Waiting For End Of User Interaction on occurrence of disconnection of the SRF (transition e6);

- from the state Waiting For End Of Temporary Connection on occurrence of disconnection of temporary connection (transition e8).

In this state the FSM for CS is waiting for an instruction from the SCF; call handling is suspended and an application timer (T_{SSF}) shall be set on entering this state.

During the state Waiting For Instructions the following events can occur:

- The user dials additional digits (applies for open-ended numbering plans): the CCF should store the additional digits dialled by the user.

- The user abandons or disconnects. This should be processed in accordance with the general rules in clause 8.2.2.

- The application Timer T_{SSF} expires: the **FSM for CS** moves to the Idle state, the CCF routes the call if possible (e.g. default routing to a terminating announcement), the T_{SSF} expiration is reported to the maintenance functions, and the operation EntityReleased is sent to the **FSM for CSA**.

- The user issues a feature activation or a Switch Hook Flash, which shall lead to the reporting of MidCall as an EDP-R or EDP-N if the Midcall DP was armed to be reported in any state except Idle.

- An operation is received from the SCF: The **FSM for CS** instance acts according to the operation received as described below.

- When an operation is received from the SCF and processed by the SSF with no resulting transition to a different state (transition e14).
  Refer to "**FSM for CS transition table**" for operations causing this transition.

  For the case where a **SplitLeg** or an **InitiateCallAttempt** (for non initial CS) operation is received from the SCF a new CS and a new instance of an **FSM for CS** instance is created by the associate **FSM for CSA**. This new FSM instance shall receive the **SplitLeg** or **InitiateCallAttempt** in the Idle state and transit to the Waiting For Instructions state (transition e3). The **FSM for CSA** also sends the **SplitLeg** operation to the source **FSM for CS** instance, which may be in the Waiting for Instructions or Monitoring state.

ReleaseCall (for a CS) or **DisconnectLeg** (for the last leg in the CS) operation may be received from the SCF. In this case, the **FSM for CS** instance shall instruct the CCF to clear the call segment and ensure that any CCF resources allocated to the call are de-allocated, processing shall continue as follows:

- if neither **CallInformationReport nor ApplyChargingReport** operation has been requested, the **FSM for CS** transits to the Idle state (transition e9);

- if **CallInformationReport or ApplyChargingReport** operation has been requested, the SSF sends each operation which has been requested from SCF, and then the **FSM for CS** transits to the Idle state (transition e9).

The **MergeCallSegments** (for source CS) operation or **MoveLeg** (for last leg in source CS) operation may be received from the SCF. In this case, the **FSM for CS** related to the source CS shall return to the Idle state (transition e9).

When processing the above operations, any necessary call handling information is provided to the call control function (CCF).

Any other operation received in this state should be processed in accordance with the general rules in 8.2.

8.2.2.3   State d: Waiting For End Of User Interaction (WFI)

The SSF enters this state from the Waiting For Instructions state (transition e5) on the reception of the operation **ConnectToResource** or the reporting of EDP-R encountered (e31).

The timer T_{SSF} is active in this state (whether it is used or not is optional).

During this state the following events can occur:

- A valid SCF-SRF operation for relaying is received and is correct, the operation is transferred to the SRF for execution.
  Refer to "**FSM for CS transition table**" for operations for relaying to SRF.

  The **FSM for CS** remains in the Waiting For End Of User Interaction(WFI) state (transition e13).

- A valid SRF-SCF event for relaying is received from SRF and is correct, the event is transferred to the SCF.
  Refer to "**FSM for CS transition table**" for events for relaying from SRF.

  The **FSM for CS** remains in the Waiting For End Of User Interaction(WFI) state (transition e13).

- The application timer T_{SSF} expires (if it was set): the **FSM for CS** moves to the Idle state, the CCF routes the call if possible (e.g. default routing to a terminating announcement), the T_{SSF} expiration is reported to the maintenance functions and the operation **EntityReleased** is sent.

- Transition to Idle state occurs immediately if operation **ReleaseCall** with '*forced release*' is requested. The '*forced release*' applies for the requested call segment(s) within the Call Segment Association.

- The user issues a feature activation or a Switch Hook Flash, which shall lead to the reporting of MidCall as an EDP-R or EDP-N if the Midcall DP was armed to be reported in any state except Idle.

- An operation is received from the SCF: The **FSM for CS** acts according to the operation received as described below.

- The user abandons or disconnects. This should be processed in accordance with the general rules in clause 8.2.2.

An operation may be received from the SCF and processed by the SSF, causing a state transition to Waiting For End Of User Interaction (Monitoring) state (transition e32).
Refer to **"FSM for CS transition table"** for operations causing this transition.

An operation may be received from the SCF and processed by the SSF with no resulting transition to a different state (transition e13).
Refer to "**FSM for CS transition table**" for operations causing this transition.

The **DisconnectForwardConnection** (only applicable for a single CS, use of this operation is not valid in a multi call segment CSA) or **DisconnectForwardConnectionWithArgument** operation may be received from the SCF and processed by the SSF in this state. Call disconnect can also be received from the SRF. In both cases this causes the release of the connection to the SRF and the transition to the Waiting For Instructions state. The disconnection is not transferred to the other party (transition e6).

Any other operation received in this state should be processed in accordance with the general rules in clause 8.2.

8.2.2.4   State e: Waiting For End Of Temporary Connection (WFI)

The FSM for CS enters this state from the Waiting For Instructions state (transition e7) upon receiving an **EstablishTemporaryConnection** operation or the reporting an EDP-R encountered (e30).

The call is routed to the assisting SSF/SRF and call handling is suspended while waiting for the end of the assisting procedure. The timer T_{SSF} is active in this state (whether it is used or not is optional).

During the state Waiting For End Of Temporary Connection the following events can occur:

- The application timer T_{SSF} expires (if it was set): the **FSM for CS** moves to the Idle state, the CCF routes the call if possible (e.g. default routing to a terminating announcement), the T_{SSF} expiration is reported to the maintenance functions and the operation **EntityReleased** is sent.

- Transition to Idle state occurs immediately if operation **ReleaseCall** with '*forced release*' is requested. The '*forced release*' applies for the requested call segment(s) within the Call Segment Association.

- The user issues a feature activation or a Switch Hook Flash, which shall lead to the reporting of MidCall as an EDP-R or EDP-N if the Midcall DP was armed to be reported in any state except Idle.

- The user abandons or disconnects. This should be processed in accordance with the general rules in clause 8.2.2.

- An operation is received from the SCF; the **FSM for CS** acts according to the operation received as described below.

An operation may be received from the SCF and processed by the SSF, causing a state transition to Waiting For End Of Temporary Connection (Monitoring) state (transition e29).
Refer to "**FSM for CS transition table**" for operations causing this transition.

Operations can be received from the SCF and processed by the SSF with no resulting transition to a different state (transition e15).
Refer to "**FSM for CS transition table**" for operations causing this transition.

The **DisconnectForwardConnection** (only applicable for a single CS, use of this operation is not valid in a multi call segment CSA) or **DisconnectForwardConnectionWithArgument** operation may be received from the SCF and processed by the SSF in this state. Call disconnect can also be received from the SRF. In both cases this causes the release of the connection to the SRF and the transition to the Waiting For Instructions state. The disconnection is not transferred to the other party (transition e8).

Any other operation received in this state should be processed in accordance with the general rules in clause 8.2.

8.2.2.5   State f: Monitoring

The FSM for CS enters this state from the Waiting For Instructions state (transition e11) upon receiving an operation when one or more EDPs are armed or/and there are other reports pending (see clause 8.2.2) or user interaction ended (e24) or Temporary connection ended (e27).
Refer to "FSM for CS transition table" for operations causing this transition.

The timer T_{SSF} is stopped on entering this state.

During the state Monitoring events can occur:
Refer to "FSM for CS transition table" regarding Events.

- If the event causing a CallInformationReport and/or ApplyChargingReport is also detected by an armed EDP-N then the **CallInformationReport** and/or ApplyCharfingReport shall be sent immediately after the corresponding **EventReportBCSM** is sent.

- If the event causing a CallInformationReport and/or ApplyChargingReport is also detected by an armed EDP-R then the **CallInformationReport** and/or ApplyChargingReport shall be sent immediately before the corresponding **EventReportBCSM** is sent.

- An operation is received from the SCF: The **FSM for CS** acts according to the operation received as described below.

- The user issues a feature activation or a Switch Hook Flash, which shall lead to the reporting of MidCall as an EDP-R or EDP-N if the Midcall DP was armed to be reported in any state except Idle.

- The user abandons or disconnects. This should be processed in accordance with the general rules in clause 8.2.2.2.

An operation can be received from the SCF and processed by the SSF with no resulting transition to a different state (transition e16).
Refer to "**FSM for CS transition table**" for operations causing this transition.

An operation may be received from the SCF and processed by the SSF, causing a state transition to the Waiting For Instructions state (transition e10, valid for SCF initiated changes of Connection view states).
Refer to "**FSM for CS transition table**" for operations causing this transition.

An operation may be received from the SCF and processed by the SSF, causing a state transition to the Idle state (transition e12, valid for SCF initiated changes of Connection view states).
Refer to **"FSM for CS transition table"** for operations causing this transition.

When the **ReleaseCall** operation is received from the SCF, the **FSM for CS** shall instruct the CCF to clear the call and ensure that any CCF resources allocated to the call are de-allocated, processing shall continue as follows:

- if neither **CallInformationReport nor ApplyChargingReport** operation has been requested, the FSM for CS transits to the Idle state (transition e12);

- if **CallInformationReport or ApplyChargingReport** operation has been requested, the SSF sends each operation which has been requested from SCF, and then the **FSM for CS** shall transit to the Idle state (transition e12).

An operation may be received from the SCF and processed by the SSF, causing a state transition to Waiting For End Of User Interaction (Monitoring) state (transition e23).
Refer to "**FSM for CS transition table**" for operations causing this transition.

An operation may be received from the SCF and processed by the SSF, causing a state transition to Waiting For End Of Temporary Connection (Monitoring) state (transition e26).
Refer to "**FSM for CS transition table**" for operations causing this transition.

Transition to Idle state occurs immediately if operation **ReleaseCall** with *'forced release'* is requested. The *'forced release'* applies for the requested call segment(s) within the Call Segment Association.

Any other operation received in this state should be processed in accordance with the general rules in clause 8.2.

8.2.2.6 State h: Waiting For End Of User Interaction (Monitoring)

The SSF enters this state from the Monitoring state (transition e23) on the reception of the operation **ConnectToResource** or Continue/ContinueWithArgument (transition e32).

The timer T_{SSF} is active in this state. (whether it is used or not is optional).

During this state the following events can occur:

- A valid SCF-SRF operation for relaying is received and is correct, the operation is transferred to the SRF for execution.
  Refer to "**FSM for CS transition table**" for operations for relaying to SRF.

- The **FSM for CS** remains in the Waiting For End Of User Interaction (Monitoring) state (transition e25).

- A valid SRF-SCF operation for relaying is received and is correct, the operation is transferred to the SCF.
  Refer to "**FSM for CS transition table**" for events for relaying from SRF.

- The **FSM for CS** remains in the Waiting For End Of User Interaction (Monitoring) state (transition e25).

- The application timer T_{SSF} expires (if it was set): the **FSM for CS** moves to the Idle state, the CCF routes the call if possible (e.g. default routing to a terminating announcement), the T_{SSF} expiration is reported to the maintenance functions and the operation **EntityReleased** is sent.

- Transition to Idle state occurs immediately if operation **ReleaseCall** with '*forced release'* is requested. The *'forced release'* applies for the requested call segment(s) within the Call Segment Association.

- An operation is received from the SCF: The **FSM for CS** acts according to the operation received as described below.

- The user abandons or disconnects. This should be processed in accordance with the rules described below.

- The following operations may be received from the SCF and processed by the SSF with no resulting transition to a different state (transition e25).
  Refer to "**FSM for CS transition table**" for operations causing this transition.

The general rules of clause 8.2.2 for event reporting are modified as follows:

- An EDP-N is reported to the SCF by sending an **EventReportBCSM** operation; the **FSM for CS** shall remain in this Waiting For End Of User Interaction (Monitoring) state (transition e25) if one or more EDPs are armed or there are report requests pending.

- If the event causing a **CallInformationReport** is also detected by an armed EDP-N then the **CallInformationReport** shall be sent immediately after the corresponding **EventReportBCSM** operation is sent.

- An EDP-R should be reported to the SCF by sending an **EventReportBCSM** operation; the **FSM for CS** should move to the Waiting For End Of User Interaction (WFI) state (transition e31).

- If the event causing a **CallInformationReport** is also detected by an armed EDP-R then the **CallInformationReport** shall be sent immediately before the corresponding **EventReportBCSM** operation is sent.

The **DisconnectForwardConnection** or **DisconnectForwardConnectionWithArgument** operation may be received from the SCF and processed by the SSF in this state. Call disconnect can also be received from the SRF. In both cases this causes the release of the connection to the SRF and the transition to the Monitoring state. The disconnection is not transferred to the other party (transition e24).

Any other operation received in this state should be processed in accordance with the general rules in 8.2.

8.2.2.7   State i: Waiting For End Of Temporary Connection (Monitoring)

The **FSM for CS** enters this state from the Monitoring state (transition e26) upon receiving an **EstablishTemporaryConnection** operation or Continue/ContinueWithArgument (transition e29).

The call is routed to the assisting SSF/SRF and call handling is not suspended while waiting for the end of the assisting procedure. The timer T_{SSF} is active in this state (whether it is used or not is optional).

During the state Waiting For End Of Temporary Connection (Monitoring) the following events can occur:

- The application timer T_{SSF} expires (if it was set): the **FSM for CS** moves to the Idle state, the CCF routes the call if possible (e.g. default routing to a terminating announcement), the T_{SSF} expiration is reported to the maintenance functions and the operation **EntityReleased** is sent.

- The receipt of an indication of disconnection of forward connection from the CCF. In this case, the SSF moves to the Monitoring state (transition e27). The disconnection is not transferred to the calling party.

- The user abandons or disconnects. This should be processed in accordance with the rules described below.

- Operations can be received from the SCF and processed by the SSF with no resulting transition to a different state (transition e28).
  Refer to "**FSM for CS transition table**" for operations causing this transition.

The general rules of clause 8.2.2 for event reporting are modified as follows:

- An EDP-N is reported to the SCF by sending an **EventReportBCSM** operation; the **FSM for CS** shall remain in this Waiting For End Of Temporary connection (Monitoring) state (transition e28) if one or more EDPs are armed or there are report requests pending.

- If the event causing a **CallInformationReport** is also detected by an armed EDP-N then the **CallInformationReport** shall be sent immediately after the corresponding **EventReportBCSM** operation is sent.

- An EDP-R should be reported to the SCF by sending an **EventReportBCSM** operation; the **FSM for CS** should move to the Waiting For End of Temporary Connection (WFI) state (transition e30).

- If the event causing a **CallInformationReport** is also detected by an armed EDP-R then the **CallInformationReport** shall be sent immediately before the corresponding **EventReportBCSM** operation is sent.

The **DisconnectForwardConnection** or **DisconnectForwardConnectionWithArgument** operation may be received from the SCF and processed by the SSF in this state. Call disconnect can also be received from the SRF. In both cases this causes the release of the connection to the SRF and the transition to the Monitoring state. The disconnection is not transferred to the other party (transition e27).

Any other operation received in this state should be processed in accordance with the general rules in clause 8.2.

8.3 Assisting SSF FSM

This clause describes the SSF FSM related to the Assisting SSF.
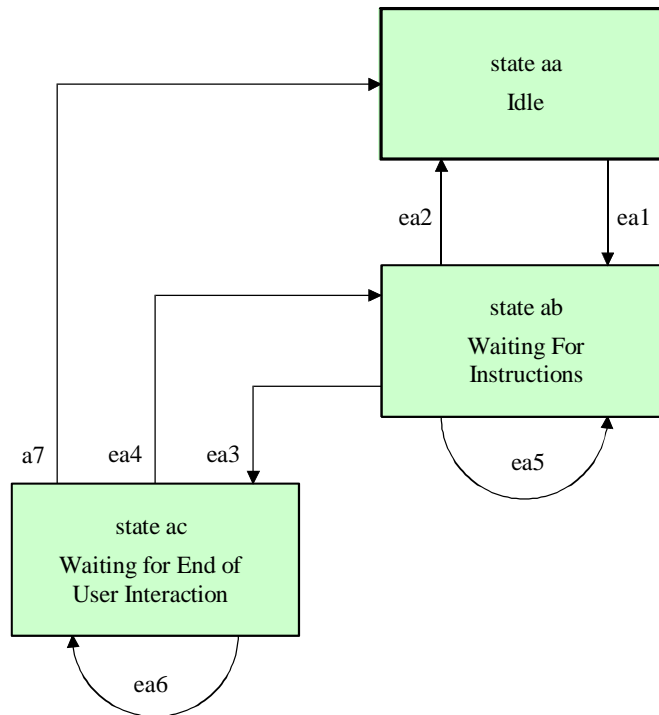
Figure 41: FSM for Assisting SSF

The Assisting SSF state diagram contains the following transitions (events):

ea1	Assist detected
ea2	Assist failed
ea3	User interaction requested
ea4	User interaction ended
ea5	Waiting For Instructions state no change
ea6	Waiting For End Of User Interaction state no change
ea7	Idle Return from Waiting For End Of User Interaction

The Assisting SSF state diagram contains the following states:

State aa	Idle
State ab	Waiting For Instructions
State ac	Waiting For End Of User Interaction

8.3.1   State aa: Idle

The **FSM for Assisting SSF** enters the Idle state when one of the following occurs:

- when sending or receiving an ABORT TCAP primitive due to abnormal conditions in any state.

- given a temporary connection between an Initiating SSF and the Assisting SSF, when a bearer channel disconnect is received from the initiating SSF; (transition ea2).

Once in the Idle state, if there are any outstanding responses to send to the SCF, they are discarded by the Assisting SSF.

The **FSM for Assisting SSF** transits from the Idle state to the Waiting For Instructions state on receipt of an assist indication at the assisting SSF from another SSF (transition ea1).

Any operation received from the SCF while the Assisting SSF is in Idle state shall be treated as an error. The event shall be reported to the maintenance functions and the transaction shall be aborted according to the procedure specified in TCAP (see ITU-T Recommendation Q.774).

8.3.2   State ab: Waiting For Instructions

This state is entered from the Idle state on receipt of a connect message at an SSF from another SSF indicating that an assist is required, based on an implementation dependent detection mechanism (transition ea1).

Before entering this state, the SSF sends an **AssistRequestInstructions** operation to the SCF and the **FSM for Assisting SSF** is waiting for an instruction from the SCF; call handling is suspended and an application timer (T_{SSF}) shall be set on entering this state.

During this state the following events can occur:

- The application timer T_{SSF} expires: the **FSM for Assisting SSF** moves to the Idle state (transition ea2) and the expiration is reported to the maintenance functions and the transaction is aborted.

- An operation is received from the SCF: The **FSM for Assisting SSF** acts according to the operation received as described below.

- A bearer channel disconnect is received and the **FSM for Assisting SSF** moves to the Idle state (transition ea2).

The following operations can be received from the SCF and processed by the Assisting SSF with no resulting transition to a different state (transition ea5):

- **ApplyCharging;**

- **FurnishChargingInformation;**

- **SendChargingInformation;**

- **ResetTimer.**

The following operations can be received from the SCF and processed by the Assisting SSF, causing a state transition to Waiting For End Of User Interaction state (transition ea3):

- **ConnectToResource.**

In the case where an implementation is not capable of differentiating between a Handed-off and an Assisting SSF case, it may execute the **ReleaseCall** operation in the assisting SSF. The **FSM for Assisting SSF** shall instruct the CCF to clear the call and ensure that any CCF resources allocated to the call are de-allocated, processing shall continue as follows:

- if **ApplyChargingReport** operation is not requested, the **FSM for Assisting SSF** transits to the Idle state (transition ea2);

- if **ApplyChargingReport** operation has been requested, the **FSM for Assisting SSF** sends **ApplyChargingReport** to SCF and then the **FSM for Assisting SSF** transits to the Idle state (transition ea2).

8.3.3   State ac: Waiting For End Of User Interaction

The **FSM for Assisting SSF** enters this state from the Waiting For Instructions state (transition ea3) on the reception of the following operation:

- **ConnectToResource.**

During this state the following events can occur:

- One of the following valid SCF-SRF operations for relaying is received and is correct, the operation is transferred to the SRF for execution:

    - **Cancel**(invokeID and optional callSegmentToCancel);

    - **PlayAnnouncement;**

    - **PromptAndCollectUserInformation;**

    - **PromptAndReceiveMessage;**

    - **ScriptClose;**

    - **ScriptInformation;**

    - **ScriptRun.**

    - The **FSM for Assisting SSF** remains in the Waiting For End Of User Interaction state (transition ea6).

- One of the following valid SRF-SCF operations for relaying is received and is correct, the operation is transferred to the SCF:

    - SpecializedResourceReport;

    - ReturnResult from PromptAndCollectUserInformation;

    - ReturnResult from PromptAndReceiveMessage;

    - ScriptEvent.

    - The **FSM for Assisting SSF** remains in the Waiting For End Of User Interaction state (transition ea6).

- The application timer T_{SSF} expires: the **FSM for Assisting SSF** moves to the Idle state, the CCF routes the call if possible (e.g. default routing to a terminating announcement), the T_{SSF} expiration is reported to the maintenance functions and the transaction is aborted (transition ea7).

- An operation is received from the SCF: The **FSM for Assisting SSF** acts according to the operation received as described below.

- A bearer channel disconnect is received from the initiating SSF and the **FSM for Assisting SSF** moves to the Idle state (transition ea7).

The following operations can be received from the SCF and processed by the SSF with no resulting transition to a different state (transition ea6):

- ResetTimer.

The **DisconnectForwardConnection** or **DisconnectForwardConnectionWithArgument** operation may be received from the SCF and processed by the Assisting SSF in this state, causing a transition to the Waiting For Instructions state (transition ea4). This procedure is only valid if a **ConnectToResource** was previously processed to cause a transition into the Waiting For End Of User Interaction state.

8.4   Handed-off SSF FSM

This clause describes the **SSF FSM** related to the **Handed-off SSF**. This SSF FSM applies only to the case where final treatment is to be applied.
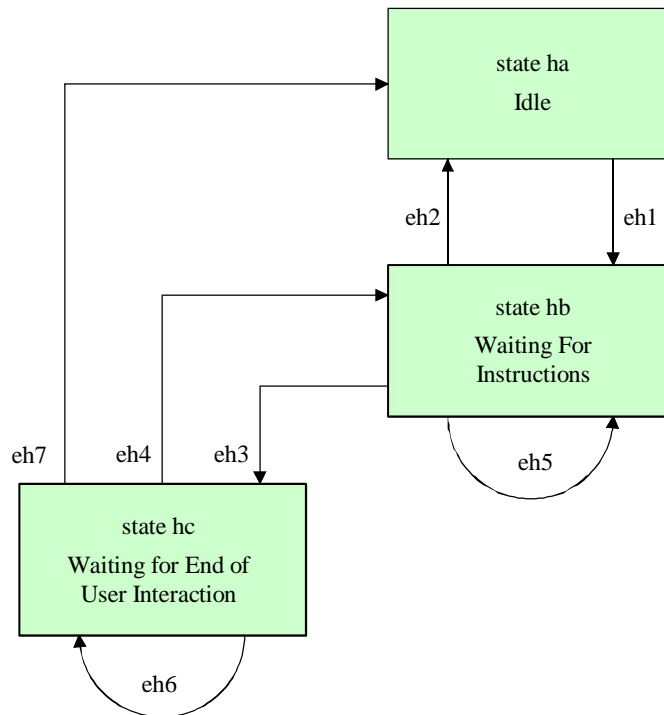
Figure 42: FSM for Handed-off SSF

The Handed-off SSF state diagram contains the following transitions (events):

eh1	Handed-off detected
eh2	Handed-off fail
eh3	User interaction requested
eh4	User interaction ended
eh5	Waiting For Instructions state no change
eh6	Waiting For End Of User Interaction state no change
eh7	Idle Return from Waiting For End Of User Interaction

The Handed-off SSF state diagram contains the following states:

State ha	Idle
State hb	Waiting For Instructions
State hc	Waiting For End Of User Interaction

8.4.1   State ha: Idle

The FSM for Handed-off SSF enters the Idle state when one of the following occurs:

- when sending or receiving an ABORT TCAP primitive due to abnormal conditions in any state.

When the bearer channel disconnects, **FSM for Handed-off FSM** shall also move to Idle.

Once in the Idle state, if there are any outstanding responses to send to the SCF, they are discarded by the Handed-off SSF.

The **FSM for Handed-off SSF** transits from the Idle state to the Waiting For Instructions state on receipt of an assist indication at the Handed-off SSF from another SSF (transition eh1).

Any operation received from the SCF while the Handed-off SSF is in Idle state should be treated as an error. The event should be reported to the maintenance functions and the transaction should be aborted according to the procedure specified in TCAP (see ITU-T Recommendation Q.774).

8.4.2   State hb: Waiting For Instructions

This state is entered from the Idle state on receipt of a connect at an SSF from another SSF indicating that an hand-off is required, based on an implementation dependent detection mechanism (transition eh1) or user interaction ended (transition eh4).

Before entering this state, the SSF sends an **AssistRequestInstructions** operation to the SCF and the **FSM for Handed-off SSF** is waiting for an instruction from the SCF; call handling is suspended and an application timer (T_{SSF}) should be set on entering this state.

During this state the following events can occur:

- The application timer T_{SSF} expires: the **FSM for Handed-off SSF** moves to the Idle state (transition eh2) and the expiration is reported to the maintenance functions and the transaction is aborted.

- An operation is received from the SCF: The **FSM for Handed-off SSF** acts according to the operation received as described below.

- A bearer channel disconnect is received and the **FSM for Handed-off SSF** moves to the Idle state (transition eh2).

The following operations may be received from the SCF and processed by the Handed-off SSF with no resulting transition to a different state (transition eh5):

- **ApplyCharging;**

- **FurnishChargingInformation;**

- **SendChargingInformation;**

- **ResetTimer.**

The following operations can be received from the SCF and processed by the Handed-off SSF, causing a state transition to Waiting For End Of User Interaction state (transition eh3):

- **ConnectToResource.**

If the **ReleaseCall** operation is received from the SCF, the **Handed-off SSF FSM** shall instruct the CCF to clear the call and ensure that any CCF resources allocated to the call are de-allocated, processing shall continue as follows:

- if **ApplyChargingReport** operation is not requested, the **Handed-off SSF FSM** transits to the Idle state (transition eh2);

- if **ApplyChargingReport** operation has been requested, the Handed-off SSF sends **ApplyChargingReport** to SCF and then the **Handed-off SSF FSM** transits to the Idle state (transition eh2).

Any other operation received in this state should be processed in accordance with the general rules in clause 8.2.

Note that multiple Handoff procedures are not covered.

8.4.3 State hc: Waiting For End Of User Interaction

The **FSM for Handed-off SSF** enters this state from the Waiting For Instructions state (transition eh3) on the reception of one of the following operations:

- **ConnectToResource.**

During this state the following events can occur:

- One of the following valid SCF-SRF operation for relaying is received and is correct, the operation is transferred to the SRF for execution:

  - **Cancel**(invokeID and optional callSegmentToCancel);

  - **PlayAnnouncement;**

  - **PromptAndCollectUserInformation;**

  - **PromptAndReceiveMessage;**

  - **ScriptClose;**

  - **ScriptInformation;**

  - **ScriptRun;**

  - The **FSM for Handed-off SSF** remains in the Waiting For End Of User Interaction state (transition eh6).

- One of the following valid SRF-SCF operations for relaying is received and is correct, the operation is transferred to the SCF:

  - SpecializedResourceReport;

  - ReturnResult from PromptAndCollectUserInformation;

  - ReturnResult from PromptAndReceiveMessage;

  - ScriptEvent;

  - The SSF for Handed-off SSF remains in the Waiting For End Of User Interaction state (transition eh6).

- When the SRF indicates to the SSF the end of user interaction by initiating disconnection the **FSM for Handed-off SSF** returns to the Waiting For Instructions state (transition eh4).

- The application timer T_{SSF} expires: the **FSM for Handed-off SSF** moves to the Idle state, the CCF routes the call if possible (e.g. default routing to a terminating announcement), the T_{SSF} expiration is reported to the maintenance functions and the transaction is aborted. (transition eh7).

- An operation is received from the SCF: The **FSM for Handed-off SSF** acts according to the operation received as described below.

The following operations can be received from the SCF and processed by the SSF with no resulting transition to a different state (transition eh6):

- **ResetTimer.**

The **DisconnectForwardConnection** or **DisconnectForwardConnectionWithArgument** operation may be received from the SCF and processed by the **Handed-off SSF** in this state, causing a transition to the Waiting For Instructions state (transition eh4). This procedure is only valid if a **ConnectToResource** was previously processed to cause a transition into the Waiting For End Of User Interaction state.

9   FSM for SCF

In case interpretations for the FSM descriptions in the following differ from detailed operation procedures and the rules for using of TCAP service, the statements and rules contained in the "Operation Procedures" in clause 11 and the "Services assumed from TCAP" in clause 15 shall be followed.

NOTE: This clause is for information only since the **FSM for the SCF** has not been fully simulated and verified.

9.1   Relationship between the SLP and the inter SCF call state model

The relationship between the SLP and the **SCF FSM** may be described as follows:

a) if a request for IN call processing is received from the SSF, an instance of an SCF Call State Model (SCSM) is created, and the relevant SLP is invoked;

b) when initiation of a call is requested from service logic, an instance of the SCSM is created.

In either case, the SCF FSM handles the interaction with other FE FSMs like e.g. the SSF FSM, and notifies the SLP of events as required.

9.2   The SCF FSM structure
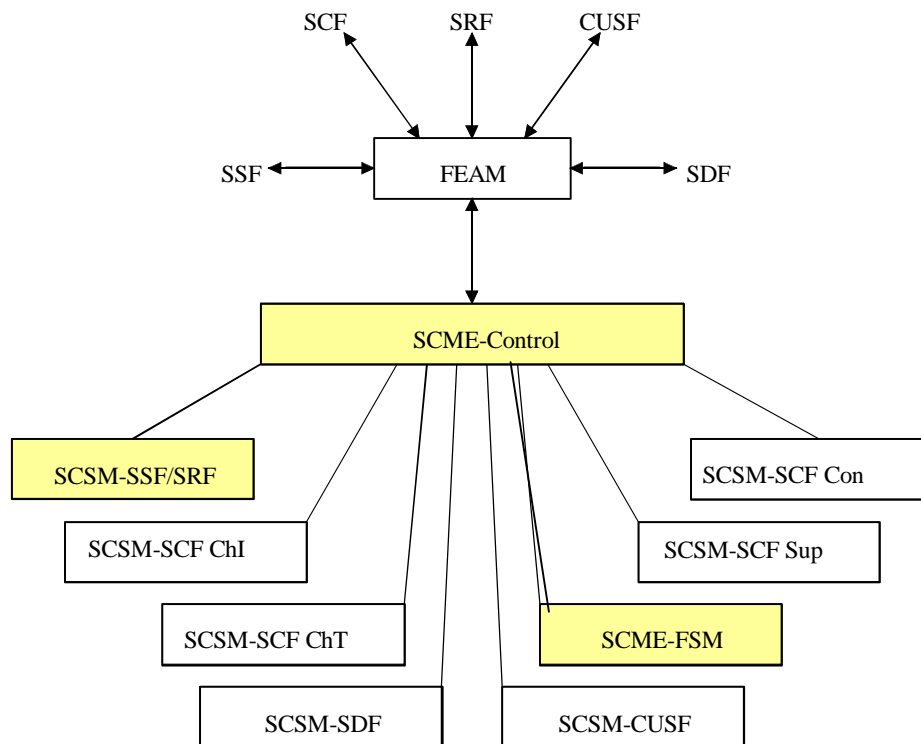
Figure 43: Inter SCF-FSM (SCSM-SCF)

For a description of the FEAM refer to EN 301 931-1.

Figure 43 shows the Inter **SCF FSM** (SCSM-SCF) structure.
The FSMs for the SCF relevant to the SCF- SSF interface are highlighted.
It includes the **SCME-Control**, **SCSM- SSF/SRF** and the **SCME-FSM** which are further described below.

The **SCME-Control** receives the ActivityTest operation from the SSF and sends the return result for the ActivityTest. The sending of ActivityTest from SSF to SCF is an option.

The relationship between the SLP and the **SCF FSM** may be described as follows:

- if a request for IN call processing is received from the SSF, an instance of an SCF Call State Model (**SCSM**) is created, and the relevant SLP is invoked;

- when initiation of a call is requested from service logic, an instance of the **SCSM** is created.

In either case, the SCF FSM handles the interaction with the SSF FSM, SRF FSM, SCF FSM, CUSF FSM, and SDF FSM, and notifies the SLP of events as required.

Notice that the interfaces shown in figure 43 are internal, and are not for standardization.

9.3   Partial SCF Management Entity (SCME) State Transition Diagram

The SCME handles the following operations:

- RequestCurrentStatusReport;

- RequestEveryStatusChangeReport;

- RequestFirstStatusMatchReport;

- CancelStatusReportRequest (including the Resource ID previously used for request FirstStatusMatchReport or RequestEveryStateChangeReport operations);

- StatusReport;

- ActivateServiceFiltering;

- ServiceFilteringResponse;

- CallGap;

- ActivityTest;

- ManageTriggerData;

- CreateOrRemoveTriggerData.

Issuing the **CallGap,** the **ManageTriggerDat**a and the **CreateOrRemoveTriggerData** operation does not cause state transitions in the SCME. The rest of the above operations are described below.

The operations that are not listed above do not affect the state of the SCME; if screening is activated these operations are screened before being passed to the relevant SCSM.

The key parts of SCF Management Entity (SCME) State Diagram are described in the following four FSMs in the SCME.

9.3.1   The Status Report FSM
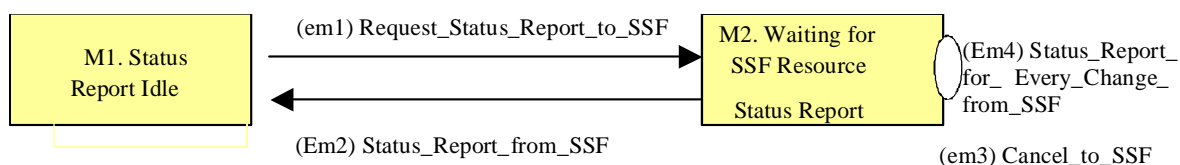
Figure 44

9.3.1.1   State M1: "Status Report Iidle"

The following event (see note) is considered in this state:

NOTE: Events are enumerated, and the number of an event is prefixed with either the letter "E" (for external
              events) or "e" (for internal ones) and included in parentheses in the beginning of the event name. The
              scope of event names and numbers is defined by the state machine in which these events appear; the same
              applies to state names.

- (em1) Request_Status_Report_to_SSF: This is an internal event, caused by a decision to transmit one of the
  following operations:

  - RequestCurrentStatusReport;

  - RequestFirstStatusMatchReport;

  - RequestEveryStatusChangeReport.

This event causes a transition to state M2, "Waiting for SSF Resource Status Report".

9.3.1.2   State M2: "Waiting for SSF Resource Status Report"

The following events are considered in this state:

- (Em2) Status_Report_from_SSF: This is an external event, caused by the reception of:

  - Return Result for the RequestCurrentStatusReport;

  - StatusReport operation for the RequestFirstStatusMatchReport; and

  - StatusReport operation for the RequestEveryStatusChangeReport caused by the expiration of the monitor
    duration timer or the cancellation of the monitor.

This event causes a transition out of this state-to-state M1, "Status Report Idle":

- (em3) Cancel_to_SSF: This is an internal event, caused by the service logic's need to end status monitoring of
  the resources in the SSF, and by transmission of CancelStatusReportRequest operation to the SSF. This event
  takes place only for previously issued RequestFirstStatusMatchReport or RequestEveryStatusChangeReport
  operations. This event causes a transition to the same state.

- (Em4) Status_Report_for_Every_Change_from_SSF: This is an external event, caused by receiving a
  StatusReport operation for reporting the resource status change in response to the
  RequestEveryStatusChangeReport operation previously issued to the SSF. This event does not cause a transition
  out of this state, so the SCME remains in state M2, "Waiting for SSF Resource Status Report".

9.3.2   The Service Filtering FSM
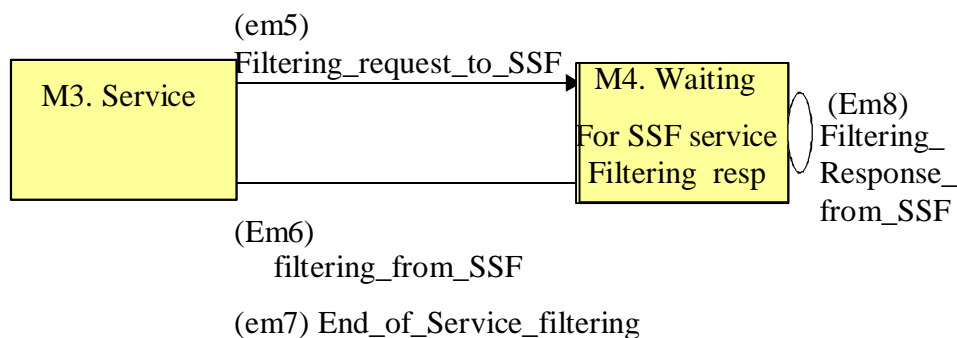
Figure 45

9.3.2.1 State M3: "Service filtering idle"

The following event is considered in this state:

- (em5) Filtering_Request_to_SSF: This is an internal event, caused by service logic's need to filter service requests to the SSF, and by transmission of the ActivateServiceFiltering operation. This event causes a transition to state M4, waiting for SSF service filtering response.

9.3.2.2 State M4: "Waiting for SSF service filtering response"

In this state, the SCF is waiting for the service filtering response from the SSF. The following events are considered in this state:

- (Em6) End_of_Service_Filtering_Response_from_SSF: This is an external event, caused by reception of the response at the end of the service filtering duration to the request service filtering previously issued to the SSF. This event causes a transition out of this state-to-state M3, service filtering idle.

- (em7) End_of_Service_Filtering: This is an internal event, caused by the expiration of service filtering duration timer in the SCF. This event causes a transition to state M3, service filtering idle.

- (Em8) Filtering_Response_from_SSF: This is an external event, caused by reception of the response to the request service filtering operation previously issued to the SSF. This event does not cause a transition out of this state, and the SCME remains in state M4, waiting for SSF service filtering response.

When service filtering is active, another service filtering operation could be sent to the SSF that has the same filtering criteria; this second "filter" replaces the first one.

9.3.3 The Activity Test FSM

(em9) Activity_Test_to_SSF

M5. Activity test idle

M6. Waiting for activity test response

(Em10) Response_from_SSF

(em11) End_of_Relationship

(Em12) TCAP_Abort
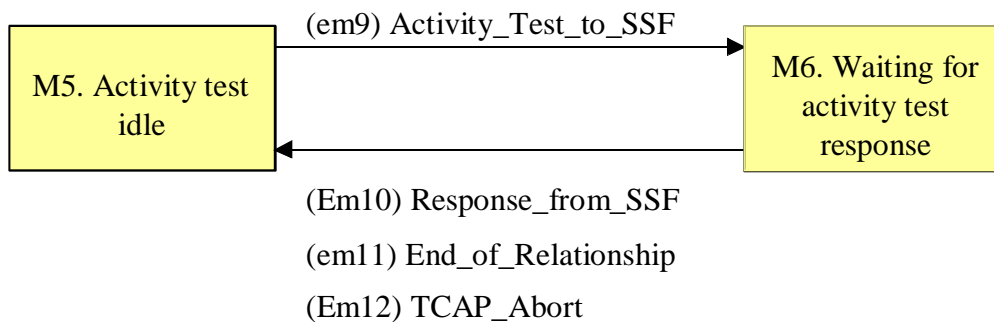
Figure 46

9.3.3.1 State M5: "Activity test idle"

The following event is considered in this state:

- (em9) Activity_test_to_SSF: This is an internal event, caused by caused by the expiration of activity test timer in the SCF, and by transmission of the ActivityTest operation. This event causes a transition to state M6, waiting for activity test response.

9.3.3.2   State M6: "Waiting for activity test response"

In this state, the SCF is waiting for the activity test response from the SSF. The following events are considered in this state:

- (Em10) Activity_Test_Response_from_SSF: This is an external event, caused by reception of the response to the activity test previously issued to the SSF. This event causes a transition out of this state-to-state M5, activity test idle.

- (em11) End_of_Relationship: This is an internal event, caused by the expiration of ActivityTest operation timer in the SCF. This event causes a transition to state M5, activity test idle.

- (Em12) TCAP_Abort: This is an external event, caused by reception of a P-Abort from TCAP in response to the ActivityTest operation previously issued to the SSF. This event causes a transition to state M5, activity test idle.

9.3.4   The Manage Trigger Data FSM
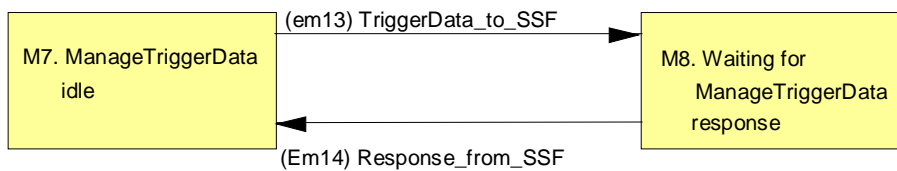
Figure 47

9.3.4.1   State M7: "ManageTriggerData idle"

The following event is considered in this state:

- (em13) TriggerData_to_SSF: This is an internal event, caused by transmission of the ManageTriggerData or CreateOrRemoveTriggerData operation. This event causes a transition to state M8, waiting for ManageTriggerData response.

9.3.4.2   State M8: "Waiting for ManageTriggerData response"

In this state, the SCF is waiting for the ManageTriggerData response from the SSF. The following events are considered in this state:

- (Em14) Response_from_SSF: This is an external event, caused by reception of the response to the ManageTriggerData or CreateOrRemoveTriggerData previously issued to the SSF. This event causes a transition out of this state-to-state M7, ManageTriggerData idle.

9.3.5   The Resource Control Object

The resource control object (RCO) is part of the SCF management entity that controls data relevant to resource information.

The RCO consists of:

1) a data structure that (by definition) resides in the SDF and can be accessed only via the RCO's methods; and

2) the RCO methods.

For purposes of the present document, no implementation constraints are placed on the structure. The only requirement to the structure is that, for each supported resource, it:

1) stores the resource's status (e.g. busy or idle); and

2) maintains the queue of SCSMs that are waiting for this resource. For continuous monitoring, the RCO maintains its knowledge of the status of the resources through use of the request every status change report operation.

The following three methods are defined for the RCO:

1) get_Resource: this method is used to obtain the address of an idle line on behalf of an SCSM. If the resource is busy, the SCSM is queued for it;

2) free_Resource: this method is used when a disconnect notification from the SSF is received. The method either advances the queue (if it is not empty) or marks the resource free (otherwise); and

3) cancel: this method is used when either the queuing timer has expired or the call has been abandoned.

9.4   SSF/SRF Related States (SCSM-SSF/SRF)

SSF/SRF related states are contained in the FSMs for SSF/SRF interface, for CSA, for Call Segment, for specialized Resource, for Handed off SSF and for Assisting SSF. The interactions between these FSMs are shown in figure 48.
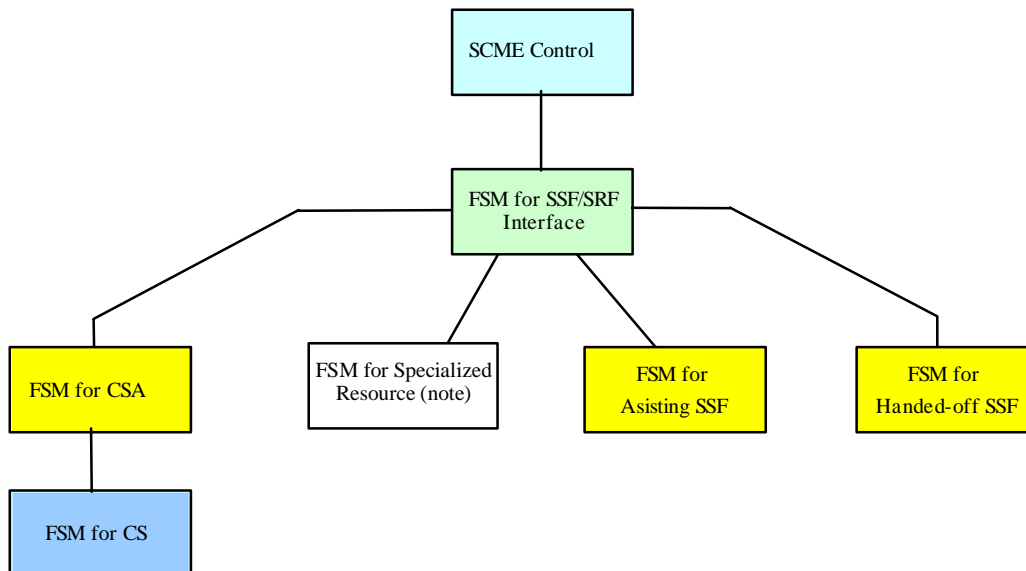
NOTE: FSM for Specialised Resource is described in ITU-T Recommendation Q.1238.3.

Figure 48: FSM interactions for SCSM-SSF/SRF

The call-control-related operations relevant to the SCF-SSF-interface (except the **SCME** related operations) are categorized into:

1) Call-processing-related operations; and

2) Non-call-processing-related operations.

Call-processing-related operations are grouped into following two sets:

- CollectInformation;

- SelectFacility;

- Connect;

- Continue;

- ContinueWithArgument;

- InitiateCallAttempt;

- ConnectToResource;

- DisconnectForwardConnection;

- DisconnectForwardConnectionWithArgument;

- EstablishTemporaryConnection;

- ReleaseCall.

For the first set of call-processing operations, the SCF may not send two operations of the same set in a series of TCAP messages or in a component sequence to the SSF, but send them only one at a time. Two operations of the first set shall be separated by at least one EDP-R message received by the SCSM. The same applies for any operation of the first set followed by ConnectToResource or EstablishTemporaryConnection.

The non-call-processing operations include the rest of the operations at the SCF-SSF interface (but not the SCME related operations). When the service logic needs to send operations in parallel, they are sent in the component sequence.

In the following, each FSM is described. The letter of "S", "I", "C", "R", "H", and "A", which are prefixed to the state numbers and the event numbers, indicates FSMs for SSF/SRF interface, for CSA, for Call Segment, for Specialized Resource, for Handed-off SSF, and for Assisting SSF, respectively.

9.4.1   Finite State Model for SSF/SRF interface

(SE3) Notification _from_SSF
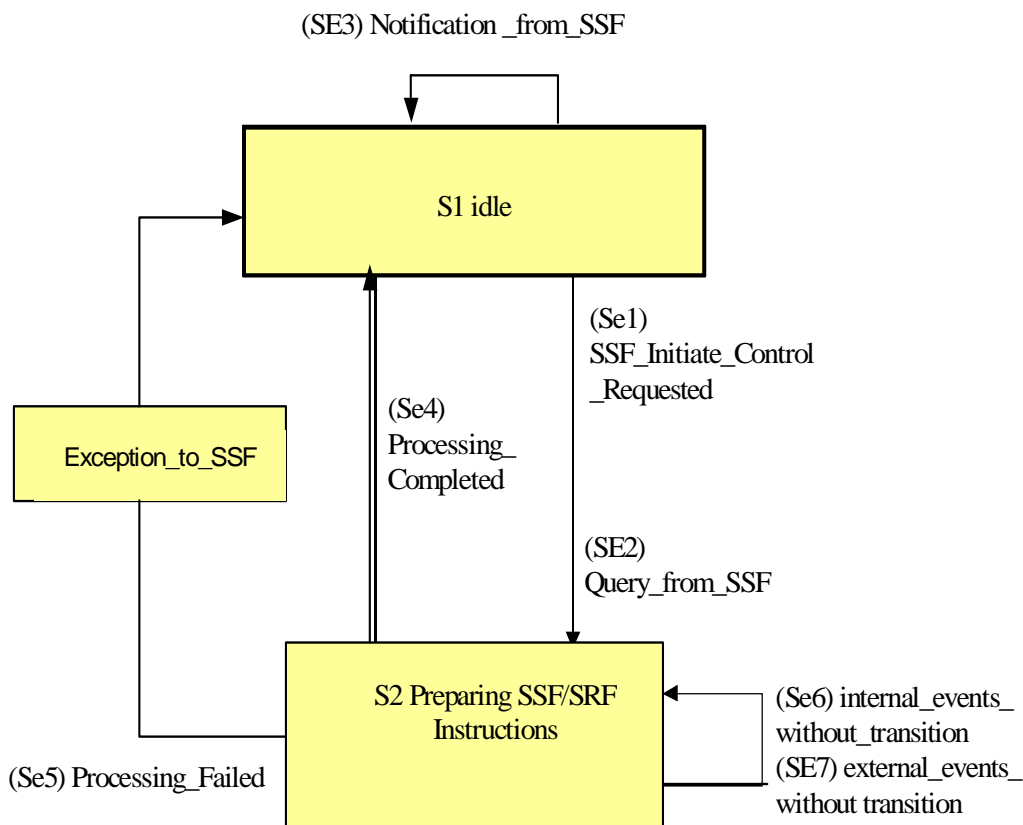
Figure 49: FSM for SSF/SRF interface

Figure 49 shows the general State Diagram of the **FSM for SSF/SRF interface** as relevant to the procedures concerning the SCF FSM part of the SCP/AD/SN during the processing of an IN call. Each state is discussed in one of the following clauses. The state of Preparing SSF/SRF Instructions has internal sub-FSMs composed of the sub-states.

The **FSM for SSF/SRF interface** has an application timer, $T_{ASSIST/HAND-OFF}$, whose purpose is to prevent excessive assist/hand-off suspension time. The FSM for SSF/SRF interface sets the timer $T_{ASSIST/HAND-OFF}$ when the SCSM sends the **EstablishTemporaryConnection** or **Connect** operation with a correlation ID. This timer is stopped when the FSM for SSF/SRF interface receives the **AssistRequestInstructions** operation from the assisting/handed-off SSF or assisting SRF. On expiration of $T_{ASSIST/HAND-OFF}$, the FSM for SSF/SRF interface informs SLPI and the maintenance functions, and the FSM for SSF/SRF interface remains in the **"Preparing SSF/SRF Instructions"** state.

9.4.1.1   State S1: "Idle"

The following events are considered in this state:

- (Se1) SSF_Initiate_Control_Requested: This is an internal event caused by the service logic's need to have a new relationship with SSF. The FSM for CSA requests to transmit the **InitiateCallAttempt** operation to the SSF. This event causes a transition to the state S2, **Preparing SSF/SRF Instructions.**

- (SE2) Query_from_SSF: This is an external event caused by a reception of one of the following operations:

  - InitialDP(for TDP-R).

This event causes a transition to the state S2, **Preparing SSF/SRF Instructions**. And the FSM for SSF/SRF interface creates a new FSM instance for CSA, and transmits this event to the FSM.
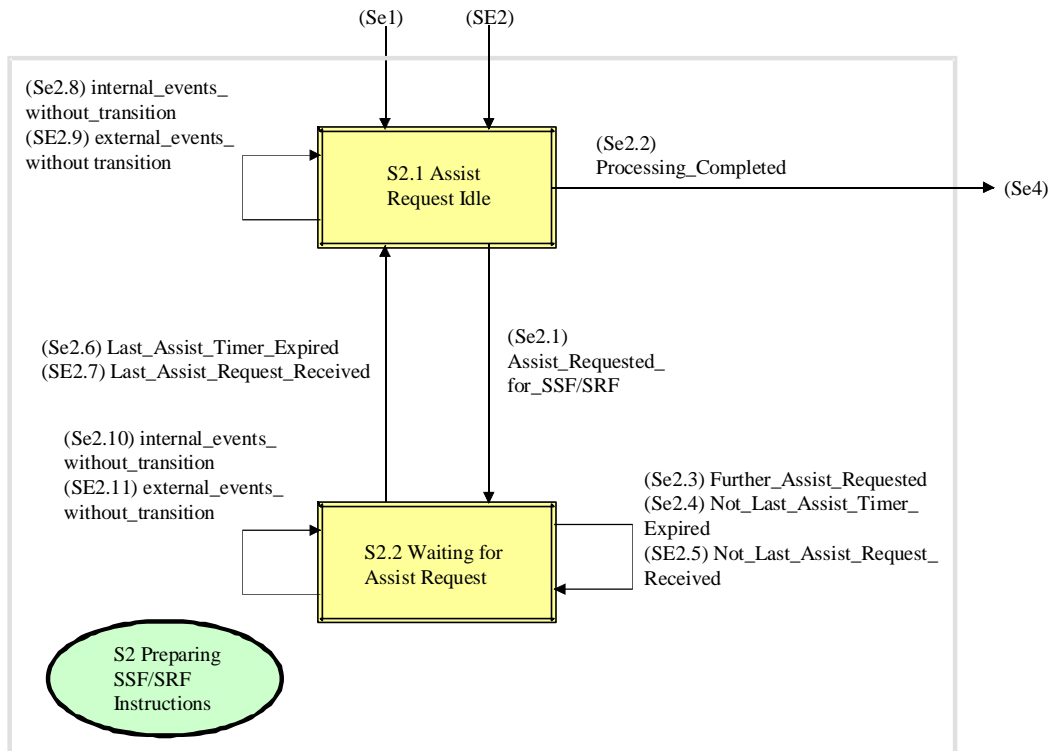
Figure 50: Partial expansion of the state S2 FSM

9.4.1.2   State S2: "Preparing SSF/SRF Instructions"

The following events are considered in this state:

- (Se4) Processing_Completed: This is an internal event caused by the end of service. In this case, the SCF has completed the processing for SSF and SRF. This event causes a transition to the state S1, **Idle**.

- (Se5) Processing_Failed: This (internal) event causes an appropriate exception processing and a transition back to the state S1, **Idle**.

  NOTE: Here and further in the present document, the exception processing is not defined. However, it is assumed that it must include releasing all the involved resources and sending an appropriate response message to the SSF. This implies that all substates handle event Se5, but it is not modelled.

- (Se6) internal_events_without_transition: This is an internal event caused by the SLPI or the other associate FSM instances. The FSM for SSF/SRF interface may send an operation to the corresponding FE. In this case, associate FSMs still exist. This event causes a transition to the same state.

- (SE7) external_events_without transition: This is an external event caused by receiving an event from the other FEs. The FSM for SSF/SRF interface will process the event and, if necessary, pass the event to the relevant associate FSMs. In this case, associate FSMs still exist. This event causes a transition to the same state.

In this state, any events received by the FSM for SSF/SRF interface which are relevant to associate FSMs are sent to those FSMs.

In this state, when all associate FSMs are released and there are no pending application timers, $T_{ASSIST/HAND-OFF}$, the FSM instance for SSF/SRF interface transits to the state 1, **Idle** and is released. However, when all associate FSMs are released and there is a pending application timer, $T_{ASSIST/HAND-OFF}$, the FSM instance for SSF/SRF interface remains in this state.

To further describe the procedures relevant to this state, this state is divided into two sub-states, which are described in the following two clauses.

9.4.1.2.1   State S2.1: "Assist Request Idle"

The following events are considered in this state:

- (Se2.1) Assist_Requested_for_SSF/SRF: This is an internal event caused by necessity of a new relationship with SSF or SRF for user interaction. In this case, SCSM sends one of the following operations to initiating SSF with an SRF address or an SSF address for routing:

  - EstablishTemporaryConnection (for Assisting SSF/SRF);

  - **Connect** (for Handed-off SSF); and

  this event causes a transition to the state S2.2, **Waiting for Assist Request.** The FSM for SSF/SRF interface starts the timer $T_{ASSIST/HAND-OFF}$.

- (Se2.2) Processing_Completed: This is an internal event. This event causes the transition that maps into the FSM event (Se4) for SSF/SRF interface.

- (Se2.8) Internal_events_without_transition: This is an internal event caused by the SLPI or the other associate FSM instances. The FSM for SSF/SRF interface may send an operation (except an **EstablishTemporaryConnection** operation and a **Connect** operation for hand-off) to the corresponding FE. In this case, associate FSMs still exist. This event causes a transition to the same state.

- (Se2.9) External_events_without transition: This is an external event caused by receiving an event from the other FEs. The FSM for SSF/SRF interface will process the event and, if necessary, pass the event to the relevant associate FSMs. In this case, associate FSMs still exist. This event causes a transition to the same state.

9.4.1.2.2   State S2.2: "Waiting for Assist Request"

In this state, the FSM for SSF/SRF waits for the **AssistRequestInstructions** operation from the Handed-off/Assisting SSF (SSF relay case) or from the SRF (Direct SCF-SRF case). The following events are considered in this state:

- (Se2.3) Further_Assist_Requested: This is an internal event caused by necessity of a new relationship with SSF or SRF for user interaction. In this case, SCSM sends one of the following operations to initiating SSF with an SRF address or an SSF address for routing:

  - EstablishTemporaryConnection (for Assisting SSF/SRF);

  - Connect (for Handed-off SSF); and

this event causes a transition to the same state. The FSM for SSF/SRF interface starts the new timer $T_{ASSIST/HAND-OFF}$.

- (Se2.4) Not_Last_Assist_Timer_Expired: This is an internal event caused by the expiration of the timer $T_{ASSIST/HAND-OFF}$ which is one of pending timers. In this case, the FSM for SSF/SRF interface informs the SLPI, and remains in the same state.

- (SE2.5) Not_Last_Assist_Request_Received: This is an external event caused by a reception of the **AssistRequestInstructions** operation. In this case, the FSM for SSF/SRF interface stops the corresponding timer, creates a new FSM instance (FSM instance for Specialized Resource, Handed-off SSF, or Assisting SSF), and transmits the event to the new FSM instance. The FSM for SSF/SRF interface remains in the same state.

- (Se2.6) Last_Assist_Timer_Expired: This is an internal event caused by the expiration of the timer $T_{ASSIST/HAND-OFF}$ which is the last pending timer. In this case, the FSM for SSF/SRF interface informs the SLPI, and transits to the state S2.1, **Assist Request Idle**. Any other pending timers are ignored.

- (SE2.7) Last_Assist_Request_Received: This is an external event caused by a reception of the **AssistRequestInstructions** operation. In this case, the FSM for SSF/SRF interface stops the corresponding timer, creates a new FSM instance (FSM instance for Specialized Resource, Handed-off SSF, or Assisting SSF), and transmits the event to the new FSM instance. The FSM for SSF/SRF interface transits to the state S2.1, **Assist Request Idle**.

- (Se2.10) internal_events_without_transition: This is an internal event caused by the SLPI or the other associate FSM instances. The FSM for SSF/SRF interface may send an operation (except an **EstablishTemporaryConnection** operation and a **Connect** operation for hand-off) to the corresponding FE. In this case, any associate FSMs which are waiting for the **AssistRequestInstructions** operation still exist. This event causes a transition to the same state.

- (SE2.11) external_events_without transition: This is an external event caused by receiving an event (except an **AssistRequestInstructions** operation) from the other FEs. The FSM for SSF/SRF interface will process the event and, if necessary, pass the event to the relevant associate FSMs. In this case, any associate FSMs which are waiting for the **AssistRequestInstructions** operation still exist. This event causes a transition to the same state.

9.4.2   Finite State Model for CSA

Figure 51 shows the general State Diagram of the FSM for CSA as relevant to the procedures concerning the SCF FSM part of the SCP/AD/SN during the processing of an IN call. Each state is discussed in one of the following clauses.
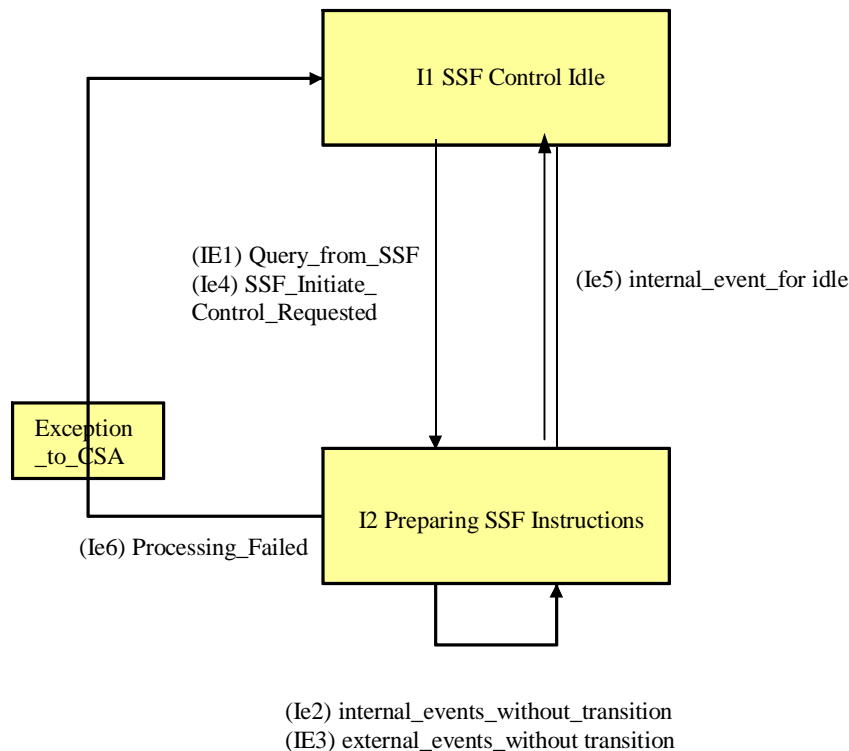
Figure 51: FSM for CSA

The **FSM for CSA** receives external events from the **FSM for SSF/SRF interface** and either processes them directly or passes them to the relevant associate **FSM for CS**. It receives internal events from the SLPI or an associate **FSM for CS** instructing it to send operations to the **FSM for SSF/SRF interface** for sending to external FEs. It will also receive internal notification of operations sent by the **FSM for the SSF/SRF interface** which affect the **FSM for CSA**.

9.4.2.1   State I1: "SSF Control Idle"

The **FSM for CSA** enters the SSF Control Idle state when one of the following occurs:

- when all the **FSM for Call Segment** instances associated with the **FSM for CSA** instance are released.

When the **FSM for CSA** enters the SSF Control Idle state, the associate **FSM for SSF/SRF interface** must be notified.

The following events are considered in this state:

- (IE1) Query_from_SSF: This is an external event caused by a reception of one of the following operations:

  - InitialDP (for TDP-R); and

this event causes a transition to the state I2, **Preparing SSF Instructions**. The **FSM for CSA** creates a new **FSM for Call Segment** instance, and transmits this event to the FSM.

- (Ie4) SSF_Initiate_Control_Requested: This is an internal event caused by the service logic's need to have a new relationship with SSF. This event occurs in the following cases.

  - The **FSM for Call Segment** requests the **FSM for CSA** to transmit the **InitiateCallAttempt** operation to the SSF.

  - The **FSM for Call Segment** requests the **FSM for CSA** to transmit the **CreateCallSegmentAssociation** operation to the SSF.

This event causes a transition to state I2, **Preparing SSF Instructions.**

9.4.2.2   State I2: "Preparing SSF Instructions"

In this state, the **FSM for CSA** instance handles instructions from the SCF and events which are received from the **FSM for CS** instances or the **FSM for SSF/SRF interface**.

The following events are considered in this state:

- (Ie2) internal_events_without_transition: This is an internal event caused by the following cases.
  When the SLPI instructs the **FSM for CSA** instance to send the following operations to the FSM for SSF/SRF interface:

  - FurnishChargingInformation;

  - Cancel(allRequests);

  - ReleaseCall;

  - MoveLeg;

  - SplitLeg.

  NOTE 1: In this case, the SSF creates a Call Segment and connects the split Leg with the Call Segment. The **FSM for CSA** transmits the event to the FSM for the 'source' CS instance. Furthermore, the **FSM for CSA** creates a new FSM for the new Call Segment instance and transmits the event to the FSM.

  - MergeCallSegments.

  NOTE 2: In this case, the SSF deletes the 'source' Call Segment and connects the Leg(s) in the 'source' Call segment with the 'target' Call Segment. The FSM for CSA transmits the event to the FSM for the 'source' CS instance and releases the FSM instance. Furthermore, the FSM for CSA transmits the event to the FSM for the 'target' CS instance.

  - RequestReportBCSMEvent.

When the FSM for SSF/SRF interface has sent the following operation:

  - MoveCallSegments.

When the associate FSM for CS instance requests the sending of the following operations:

- ApplyCharging;

- CallInformationRequest;

- RequestNotificationChargingEvent;

- SendChargingInformation;

- ResetTimer;

- Cancel(invokeID and optional callSegmentToCancel);

- ConnectToResource;

- EstablishTemporaryConnection;

- DisconnectForwardConnection;

- DisconnectForwardConnectionWithArgument;

- PlayAnnouncement;

- PromptAndCollectUserInformation;

- PromptAndReceiveMessage;

- InitiateCallAttempt;

- Connect;

- CollectInformation;

- ScriptInformation;

- ScriptRun;

- ScriptClose;

- SelectFacility;

- Continue;

- ContinueWithArgument;

- Release Call;

- DisconnectLeg;

- SetServiceProfile.

When the application timer Tassist/hand-off in the **FSM for SSF/SRF interface** expires.

In this case, any associate FSMs still exist. This event causes a transition to the same state.

- (IE3) external_events_without transition: This is an external event caused by receiving an event from the other FEs. The **FSM for CSA** will process the event and, if necessary, pass the event to the relevant associate FSMs.

    - EventReportBCSM;

    - CallInformationReport;

    - ApplyChargingReport;

    - EntityReleased;

    - EventNotificationCharging;

- SpecializedResourceReport;

- ReturnResult from PromptAndReceiveMessage;

- ScriptEvent;

- ReturnResult for PromptAndCollectUserInformation.

In this case, any associate FSMs are existing. This event causes a transition to the same state.

- (Ie5) internal_event_for idle: This is an internal event caused by the following cases.

    When the last FSM instance for CS transits to the idle state.

In this case, any associate FSMs are not existing. This event causes a transition to the state I1, **SSF Control Idle**.

- (Ie6) Processing_Failed: This (internal) event causes an appropriate exception processing and a transition to the state I1, **SSF Control Idle**.

9.4.3   Finite State Model for Call Segment

Figure 52 shows the general State Diagram of the **FSM for Call Segment** as relevant
to the procedures concerning the **SCF FSM** part of the SCP/AD/SN during the processing of an IN call. Each state is discussed in one of the following clauses.

The following operations (the CPH operations) are of class 1 and require reception of a Return Result:

- DisconnectLeg;

- MergeCallSegments;

- MoveCallSegments;

- MoveLeg;

- SplitLeg.

Reception of the Return Result for the operations mentioned above does not result in a state transition in the **FSM for CS**. Furthermore, this Return Result may be received in any state of the **FSM for CS**.

Reception of Return Result may support keeping an accurate CV in SCF and decide what subsequent action can be taken by the service logic. If various components (e.g. including CPH operations) are grouped into a single TC message the Return Result will provide the invoke identifier. This allows the service logic to correlate the result to a previously sent CPH operation and hereby distinguish between the grouped components (operations) sent.

The **FSM for CS** has an application timer, $T_{SCF-SSF}$, whose purpose is to restart the timer, T_{SSF}, to guard the release of the call segment which is waiting for instructions from the SCF. The use of timer $T_{SCF-SSF}$ for the **FSM for CS** is mandatory.

The timer $T_{SCF-SSF}$ is set in the following cases:

i) *when the SCF receives an **InitialDP** operation. In this case, this timer is restart when a first request, other than **ResetTimer** operation, is sent to the SSF. On the expiration of timer $T_{SCF-SSF}$, the **FSM for CS** may restart T_{SSF} once using the **ResetTimer** operation, and restart timer $T_{SCF-SSF}$. On the second expiration of $T_{SCF-SSF}$, the **FSM for CS** informs the SLPI and the maintenance functions, and the **FSM for CS** transitions to the state C1, **CS Control Idle**;*

ii) *when the **FSM for CS** instance enters the **Preparing CS Instructions** state(C2.1) under any other condition than the case i). In this case, the **FSM for CS** may restart T_{SSF} using the **Reset Timer** operation any number of times;*

*iii) when the **FSM for CS** enters the Queuing substate (see clause 2.5.10.3.2.3, State C2.3: "**Queuing**"). In this case, on the expiration of timer $T_{SCF-SSF}$, the **FSM for CS** may restart T_{SSF} using the **ResetTimer** operation any number of times; and*

*iv) when the SCF enters the **Suspended and User Interaction** state (C3). In this case, on the expiration of timer $T_{SCF-SSF}$, the **FSM for CS** may restart T_{SSF} using the **ResetTimer** operation any number of times.*

In each of the cases, $T_{SCF-SSF}$ may have different values as defined by the application. The values of $T_{SCF-SSF}$ are smaller than the respective values of T_{SSF}.

When receiving or sending any other operation, the SCF should restart $T_{SCF-SSF}$. Whenever the value of is changed by the SCF sending a **ResetTimer** operation to the SSF, the value of $T_{SCF-SSF}$ must be changed accordingly.
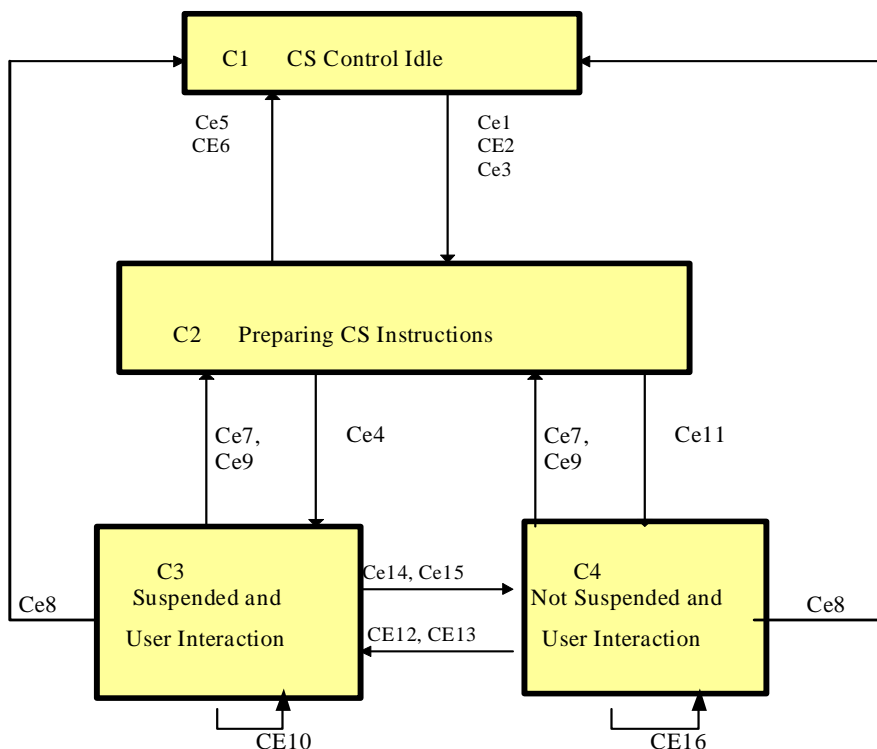
Figure 52: FSM for Call Segment

The SCF state diagram for the CS contains the following transitions (events):

Ce1   New_Call_Segment_from_SLPI
CE2   Query_from_SSF
Ce3   New_CS_Creation_for_Split
Ce4   SR_Fac._Needed (Call Processing Suspended)
Ce5   Processing_Completed
CE6   Last_Event_Received
Ce7   Continue_SCF_Processing
Ce8   Processing_Completed
Ce9   MidCall EDP-R/N (report in any state)
CE10   CS_Monitoring_Needed
Ce11   SR_Facilities_Needed (Call Processing not Suspended)
CE12   CS_Instruction_Needed_During _Established_Temporary_Connection
CE13   CS_Instruction_Needed_During_User_Interaction
Ce14   CS_Monitoring_Needed_During_Established_Temporary_Connection
Ce15   CS_Monitoring_Needed_During_User_Interaction
CE16   MidCall EDP-N

9.4.3.1 State C1: "CS Control Idle"

The **FSM for CS** must enter the state C1 **CS Control Idle** when the associate **FSM for CSA** instance transits to the state I1 **SSF Control Idle**.

When the **FSM for CS** enters the state **CS Control Idle**, the associate **FSM for CSA** must be notified.

The following events are considered in this state:

- (Ce1) New_Call_Segment_from_SLPI: this is an internal event caused by the SLPI when there is a need to send the following operation to the SSF.

    - InitiateCallAttempt.

This event causes a transition to the state C2, **Preparing CS Instructions**.

- (CE2) Query_from_SSF: this is an external event caused by a reception of one of the following from the SSF.

    - InitialDP.

This event causes a transition to the state C2, **Preparing CS Instructions**.

- (Ce3) New_Call_Segment_for_Split: this is an internal event caused by the SLPI when there is a need to send the following operation to the SSF.

    - SplitLeg (target CS).

This event causes a transition to the state C2, **Preparing CS Instructions**.

9.4.3.2 State C2: "Preparing CS Instructions"

The following events are considered in this state:

- (Ce4) SR_Facilities_Needed (Call Processing Suspended): this is an internal event caused by the service logic's need for additional information from the call party; hence is the necessity to set up a connection between the call party and the SRF. This event causes a transition to the state C3, **Suspended and User Interaction**.

- (Ce5) Processing_Completed: this is an internal event, caused by the end of processing for the CS. This event causes a transition to the state C1, **CS Control Idle**.

- (CE6) Last_Event_Received: this is an external event caused by a reception of a last event from the SSF. This event causes a transition to the state C1, **CS Control Idle**.

- (Ce11) SR_Facilities_Needed (Call Processing not Suspended): this is an (internal) event caused by the service logic's need for additional information exchange with the call party, while waiting for request or notification from the SSF. There is therefore the necessity to set up a connection between the call party and the SRF. This event causes a transition to the state C4, **Not Suspended and User Interaction.**

In order to further describe the procedures relevant to this state, the state is divided into three sub-states, which are described in the following. This subdivision is illustrated in figure 53.

Figure 53: Partial expansion of the state C2 FSM for CS
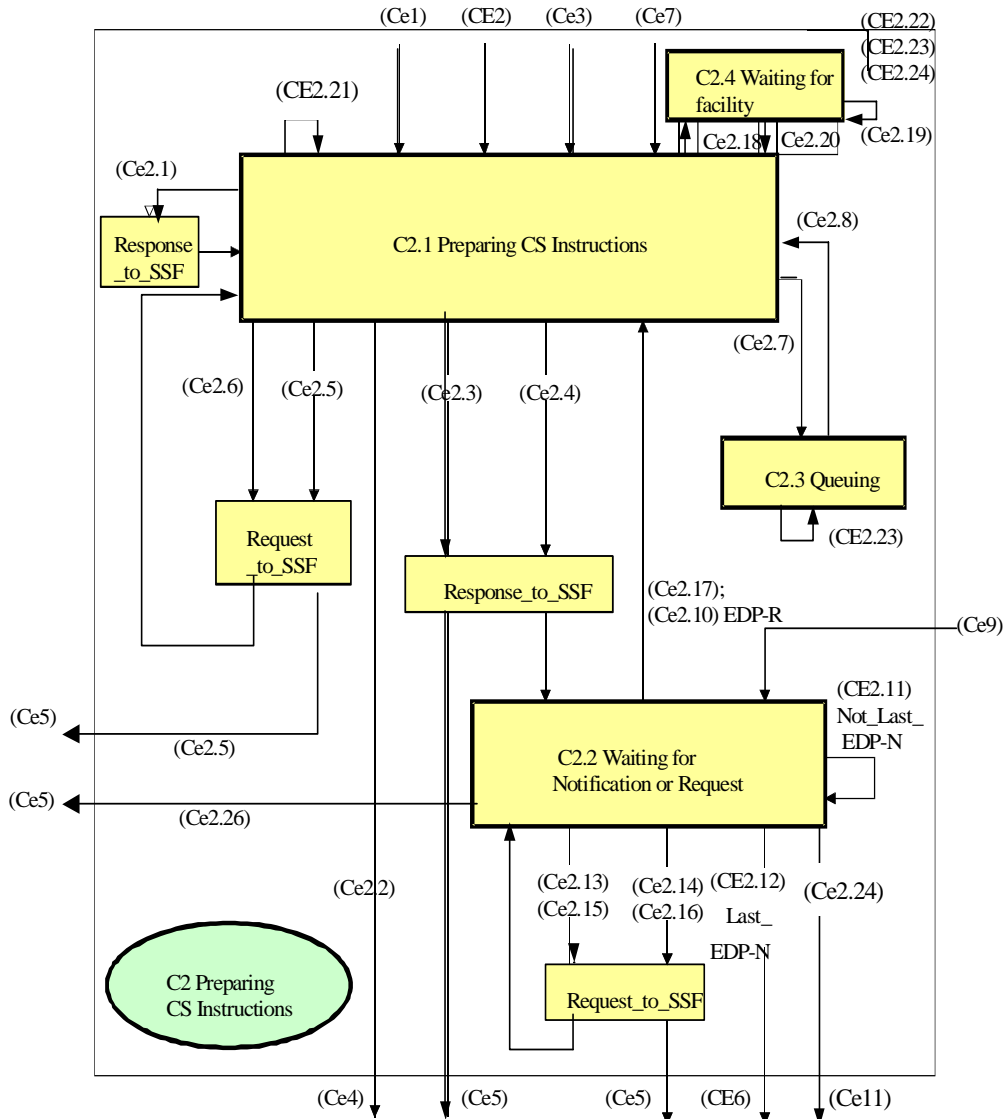
(Ce2.3):	Call_Processing_Instruction_Ready (Monitoring* not Required)
(Ce2.4):	Call_Processing_Instruction_Ready (Monitoring* Required)
(Ce2.5), (Ce2.26):	CS_or_Leg_Control_Last_Instruction
(Ce2.6), (Ce2.17):	CS_or_Leg_Control_Continuing_Instruction
(Ce2.7):	Queue_Processing_Needed
(Ce2.8):	Queue_Processing_Finished
(Ce2.13):	Notification_or_Request_Continuing_Instruction
(Ce2.14):	Monitoring_Cancel_Instruction
(Ce2.15):	Release_Call_Instruction (Call Information Report or Apply Charging Report has been requested)
(Ce2.16):	Release_Call_Instruction (Neither CallInformationReport nor ApplyChargingReport has been requested)
(Ce2.18):	First_Publicity_Events
(Ce2.19):	Subsequent Facility Event
(CE2.20):	Last_Facility_Event
(CE2.21), (CE2.22):	MidCall_EDP
(CE2.23):	Not_Last_Facility_Event
(Ce2.2), (Ce2.24):	SR_Facilities_Needed
(Ce2.25):	Non-Call_Processing_Instructions

Event numbers prefixed with "E" are external.
Event numbers prefixed with "e" are internal.

9.4.3.2.1 State C2.1: "Preparing CS Instructions"

The following events are considered in this state:

- (Ce2.1) Non-Call_Processing_Instructions: this is an internal event caused by the following cases.

  - When the service logic needs to send an operation such as the following to the SSF;

  - ApplyCharging;

  - CallInformationRequest;

  - RequestNotificationChargingEvent;

  - SendChargingInformation;

  - SetServiceProfile; and

  - FurnishChargingInformation.

When one of the following operations has been sent to the SSF by an FSM for CSA:

  - Cancel (allRequests);

  - RequestReportBCSMEvent.

When the application timer $T_{SCF-SSF}$ expires, the FSM for CS sends a ResetTimer operation to the corresponding CS FSM in the SSF.

This event causes a transition back to state C2.1, **Preparing CS Instructions**.

- (Ce2.2) SR_Facilities_Needed: this is an internal event, caused by the service logic when there is a need to use the SRF. This event is mapped as the FSM event (Ce4).

- (Ce2.3) Call_Processing_Instruction_Ready (monitoring not required): this is an internal event caused by the service logic when the final call-processing-related operation is ready and there is neither an armed EDP nor an outstanding **CallInformationReport** or **ApplyChargingReport** operation. It causes one of the following operations to be issued to the SSF:

  - CollectInformation;

  - Connect;

  - Continue.

(Only applicable for a single CS with no more than two legs. Use of this operation is not valid in a multi call segment CSA.)

  - ContinueWithArgument;

  - ReleaseCall;

  - SelectFacility.

In addition, one or more of the following operations may be issued to the SSF prior to the operations listed above:

  - Cancel (allRequests);

  - RequestReportBCSMEvent (to disarm all the armed EDPs);

  - RequestNotificationChargingEvent
    (to set mode transparent to stop monitoring for charging events); and

  - SendChargingInformation.

This event is mapped as the FSM event (Ce5).

- (Ce2.4) Call_Processing_Instruction_Ready (monitoring required): this is an internal event caused by the service logic when a call-processing-related operation is ready and the monitoring of the call is required (e.g. an EDP is armed, or there is an outstanding **CallInformationReport** or **ApplyChargingReport**). It causes one of the following operations to be issued to the SSF:

    - CollectInformation;

    - Connect;

    - Continue.

(Only applicable for a single CS with no more than two legs. Use of this operation is not valid in a multi call segment CSA.):

    - ContinueWithArgument;

    - ReleaseCall;

    - SelectFacility.

In addition, one or more of the following operations may be issued to the SSF prior to the operations listed above:

    - ApplyCharging;

    - CallInformationRequest;

    - RequestReportBCSMEvent;

    - RequestNotificationChargingEvent;

    - SendChargingInformation;

    - SetServiceProfile.

This event causes a transition to the state C2.2, **Waiting for Notification or Request**.

- (Ce2.5) CS_or_Leg_Control_Last_Instruction: this is an internal event caused by the following cases.

When the SLPI needs to send an operation such as the following to the SSF:

    - **DisconnectLeg** (for last leg).

When one of the following operations has been sent to the SSF by an **FSM for CSA** instance:

    - **MergeCallSegments** (for 'source' CS);

    - **MoveLeg** (for last leg in 'source' CS).

This event is mapped as the FSM event (Ce5).

- (Ce2.6) CS_or_Leg_Control_Continuing_Instruction: this is an internal event caused by the following cases.

When the SLPI needs to send an operation such as the following to the SSF:

    - **DisconnectLeg** (for not last leg).

When one of the following operations has been sent to the SSF by an **FSM for CSA** instance:

    - **MergeCallSegments (for 'target' CS);**

    - SplitLeg (for 'source' CS);

    - MoveLeg (for 'target' CS or not last leg in 'source' CS);

    - MoveCallSegments.

This event causes a transition back to the same state.

- (Ce2.7) Queuing_Processing_Needed: this is an internal event caused by the service logic when queuing of the call is required. This event causes a transition into the state C2.3, **Queuing**.

- (CE2.21) MidCall_EDP: This is an external event caused by a reception of one of the following operations if O/T-MidCall EDP-R/N was armed to be reported in "any state":

  - EventReportBCSM for O/T-MidCall.

This event causes a transition back to the same state.

9.4.3.2.2   State C2.2: "Waiting for Notification or Request"

The following events are considered in this state:

- (Ce2.10) EDP-R: this is an external event, caused by a reception of one of the following operations:

  - EventReportBCSM (for EDP_R).

This event causes a transition to the state C2.1, **Preparing CS Instructions**.

- (CE2.11) Not_Last_EDP-N: this is an external event, caused by a reception of one of the following operations:

  - ApplyChargingReport;

  - CallInformationReport;

  - EventReportBCSM (for EDP_N);

  - EventNotificationCharging.

In this case, there is still an outstanding armed EDP or pending **CallInformationReport** or **ApplyChargingReport** operation. This event causes a transition back to the state C2.2**, Waiting for Notification or Request**.

Refer to the meaning of 'last EDP-N' which is described in the event (CE2.12).

- (CE2.12) Last_EDP-N: this is an external event, caused by a reception of one of the following operations:

  - ApplyChargingReport;

  - CallInformationReport;

  - EntityReleased;

  - EventReportBCSM (for EDP_N).

In this case, there is no outstanding armed EDP and neither pending **CallInformationReport** nor pending **ApplyChargingReport**. This event is mapped as the FSM event (CE6).

   NOTE: The 'last EDP-N' means that there are no other EDPs which may be encountered when an EDP-N was detected.
             Some of the EDPs are automatically disarmed if another EDP is encountered. The EDPs which are automatically disarmed depend on which EDP is encountered.
             An example is the case of the EDPs O_Answer, O_No_Answer, RouteSelectFailure or O_Called_Party_Busy. If any of these EDPs are encountered, all the other EDPs of this list are automatically (implicitly) disarmed.

- (Ce2.13) Notification_or_Request_Continuing_Instruction: this event is an internal event caused by the following cases.

  When the SLPI needs to send an operation such as the following to the SSF.

  - ApplyCharging;

  - FurnishChargingInformation;

  - RequestNotificationChargingEvent;

- SendChargingInformation;

- SetServiceProfile.

When the following operation has been sent to the SSF by the FSM for CSA**:**

- RequestReportBCSMEvent (for the purpose of which

    i) a part of armed EDPs will be disarmed; or

    ii) all of armed EDPs will be disarmed when there are other pending requests).

This event causes a transition back to the state C2.2, **Waiting for Notification or Request**.

- (Ce2.14) Monitoring_Cancel_Instruction: this is an internal event caused when the **FSM for CSA** instance has sent one of the following operations to the SSF:

- RequestReportBCSMEvent (for the purpose of which all of armed EDPs will be disarmed when there are no more other pending requests); and

- Cancel (AllRequests).

This event is mapped as the FSM event (Ce5).

- (Ce2.15) Release_Call_Instruction (Call Information Report or Apply Charging Report has been Requested): this is an internal event caused when the **FSM for CSA** instance has sent the following operation to the SSF:

- ReleaseCall **(when there is outstanding** CallInformationReport **or** ApplyChargingReport**).**

This event causes a transition back to the state C2.2, **Waiting for Notification or Request**.

- (Ce2.16) Release_Call_Instruction (neither Call Information Report nor Apply Charging Report has been Requested): this is an internal event caused when the **FSM for CSA** instance has sent the following operation to the SSF:

- ReleaseCall **(when there is no outstanding** CallInformationReport **or** ApplyChargingReport**).**

This event is mapped as the FSM event (Ce5).

- (Ce2.17) CS_or_Leg_ Control_Continuing_Instruction: this is an internal event caused by the following case.

- When the SLPI needs to send an operation such as the following to the SSF:

    - **DisconnectLeg** (for not last leg).

When the following operations have been sent to the SSF by the **FSM for CSA** instance:

- MergeCallSegments (for 'target' CS);

- MoveLeg (for 'target' CS and not last leg in 'source' CS);

- SplitLeg.

This event causes a transition back to the state C2.1, Preparing CS Instructions.

- (Ce2.24) SR_Facilities_Needed: this is an internal event, caused by the service logic when there is a need to use the SRF. This event is mapped as the FSM event (Ce11).

- (Ce2.26) CS_or_Leg_Control_Last_Instruction: this is an internal event, caused by the following cases:

- When the SLPI needs to send an operation such as the following to the SSF:

    - DisconnectLeg **(for last leg).**

When the following operations have been sent to the SSF by an **FSM for CSA** instance:

- MergeCallSegments (for 'source' CS);

- MoveLeg (for last leg in 'source' CS).

This event is mapped as the FSM event (Ce5).

9.4.3.2.3   State C2.3: "Queuing"

When the SCF is processing the query from the CCF/SSF, it may find that the resource to which the call shall be routed is unavailable. One possible reason causing the resource to be unavailable is the "busy" condition.

Such a resource may be an individual line or trunk or a customer-defined group of lines or trunks. In the latter case, the word "busy" means that all lines or trunks in the group are occupied; and the word "idle" means that at least one line or trunk in the group is idle.

If the resource is busy, the SCF may put the call on queue and resume it later when the resource is idle.

The following events are considered in this state:

- (Ce2.8) Queuing_Processing_Finished: This is an internal event caused by the SLP when it is ready to prepare the call-related operation for sending to the SSF. This event causes a transition to State C2.1, **Preparing CS Instructions**.

This state further expands into an FSM, which is depicted in figure 54.

This FSM does not explicitly describe all possible combinations of resource monitoring functions used for queuing. The following possibilities may be used in implementations:

- RequestFirstStatusMatchReport **by means of the SCME;**

- RequestCurrentStatusReport **by means of the SCME;**

- RequestEveryStatusChangeReport **by means of the SCME; and**

- **monitoring based on issuing by the SCSM of the** RequestReportBCSMEvent **operation and subsequent reception of the** EventReportBCSM **operation to report the availability of the resource. Both the Request and Report occur in a single different call context. In this case, operations to the SDF or equivalent SCF functionality may be used for scanning the status of resources.**

- (CE2.23) MidCallEDP: this is an external event caused by a reception of. one of the following operations if O/T-MidCall EDP-R/N was armed to be reported in "any state":

- **EventReportBCSM** for O/T-MidCall.

This event causes a transition back to the same state.
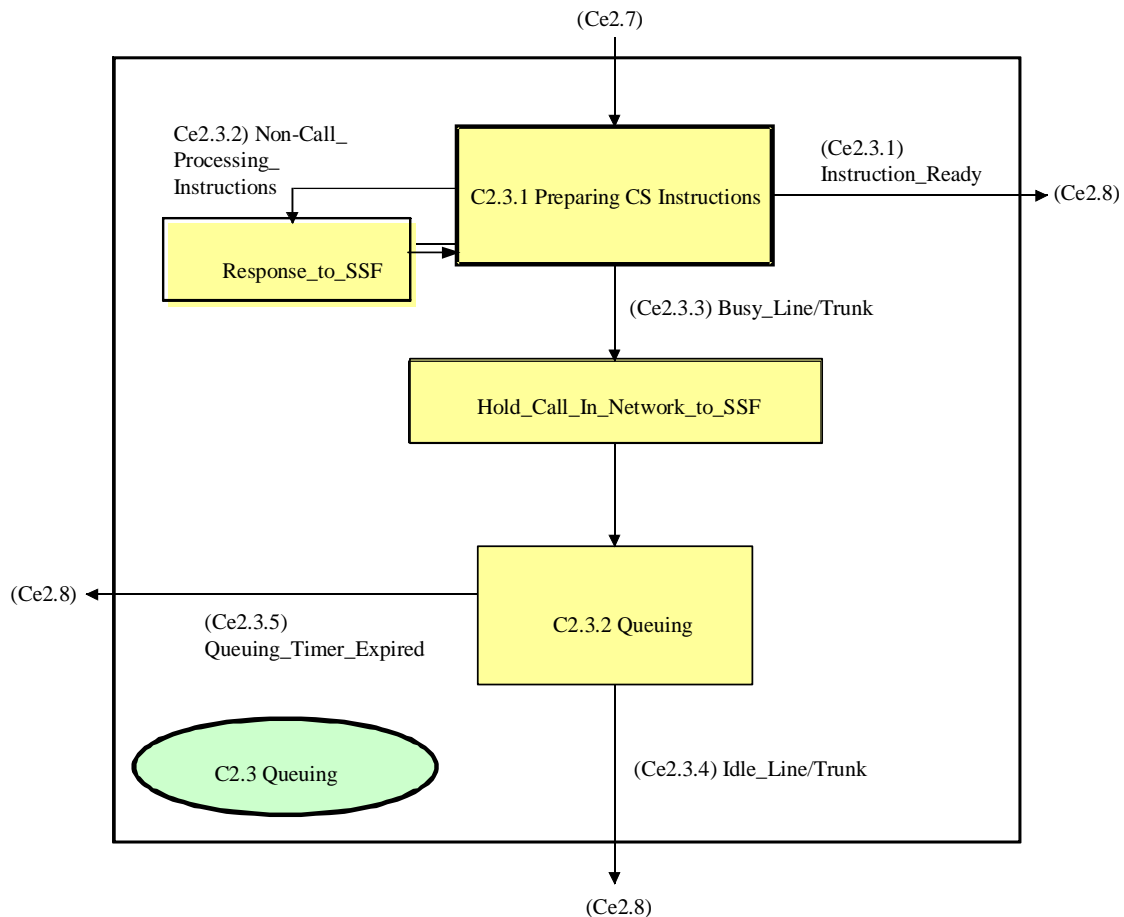
Figure 54: Partial expansion of the state C2.3 FSM for CS

9.4.3.2.3.1   State C2.3.1: Preparing CS Instructions

In this state, the FSM for CS prepares the instructions for the SSF to complete the call. The following events are considered in this state:

- (Ce2.3.1) Instruction_Ready: this is an internal event that takes place only when the required resource is available. In this case, the **FSM for CSA** has obtained the address of the free resource via the Get_Resource method of the Resource Control Object, RCO (see clause 9.3.5). This event maps into the event (Ce2.8).

- (Ce2.3.2) Non-Call_Processing_Instructions: This is an internal event caused by the following cases.

When the service logic needs to send an operation such as the following to the SSF:

- ApplyCharging;

- CallInformationRequest;

- FurnishChargingInformation;

- RequestNotificationChargingEvent;

- SendChargingInformation;

- SetServiceProfile.

When the following operations have been sent to the SSF by the FSM for CSA instance:

- RequestReportBCSMEvent.

- When the application timer $T_{SCF-SSF}$ expires, the FSM for CS sends a **ResetTimer** operation to the SSF for the corresponding CS.

This event causes a transition back to the state C2.3.1, **Preparing CS Instructions**.

- (Ce2.3.3) Busy_Line/Trunk: this is an internal event caused by the RCO when no terminating line/trunk is available. This event causes the HoldCallInNetwork operation to be sent to the SSF, and a transition to the state C2.3.2, **Queuing**.

9.4.3.2.3.2 State C2.3.2: "Queuing"

In this state, the FSM for CS is awaiting an indication from the RCO to proceed with routing a call to an idle trunk/line. The support of playing various announcements is also provided when the FSM for CS is in this state. In the present document, the relevant further expansion of the state is not provided; however, it is not different from that of the state C3. Nevertheless, if announcements are completed before the call is de-queued and the SSF FSM has transitioned to the state **Waiting for Instructions**, the operation should be sent to set the T_{SSF} with an appropriate value. Once the FSM for CS enters this state, the Queuing Timer is started. The role of this timer is as follows:

- the Queuing Timer limits the time that a call can spend in the queue, and its value may be customer-specific.

The following events are considered in this state:

- (Ce2.3.4) Idle_Line/Trunk: this is an internal event, which maps into the event (Ce2.8).

- (Ce2.3.5) Queuing_Timer_Expired: this is an internal event, which results in processing the Cancel method of the RCO and maps into the event (Ce2.8) (following procedures depends on the decision of the service logic that may play (or not play) the terminating announcement).

9.4.3.3 State C3: "Suspended and User Interaction"

The following events are considered in this state:

- (Ce7) Continue_SCF_Processing: in this case, the SCF has obtained all the information from the SRF, that is needed to instruct the SSF to complete the call. This event causes a transition to state C2.1, **Preparing CS Instructions**.

- (Ce8) Processing_Completed: this is an internal event, caused by the end of processing for the CS. This event causes a transition to the state C1, **CS Control Idle**.

- (Ce9) CS_Monitoring_Needed: this is an internal event caused by the necessity of monitoring. This event causes a transition to the state C2.2, **Waiting for Notification or Request.**

- (CE10) MidCall_EDP: this is an external event caused by a reception of one of the following operations if O/T-MidCall EDP-R/N was armed to be reported in "any state":

  - **EventReportBCSM** (for O/T-MidCall).

This event causes a transition back to the same state.

- (Ce14) CS_Monitoring_Needed_During_Established_Temporary_Connection: this is an internal event caused by the necessity of monitoring, while being in state C3.3 Establishing Temporary Connection. This event causes a transition to the state C4.3, **Establishing Temporary Connection Monitoring**.

- (Ce15) CS_Monitoring_Needed_During_User_Interaction: this is an internal event caused by the necessity of monitoring, while being in the state C3.2 User Interaction. This event causes a transition to the state C4.2, **User Interaction Monitoring**.

To further describe the procedures relevant to this state, the state is divided into three sub-states, which are described in the following three clauses. This subdivision is illustrated in figure 55.

(Ce4)
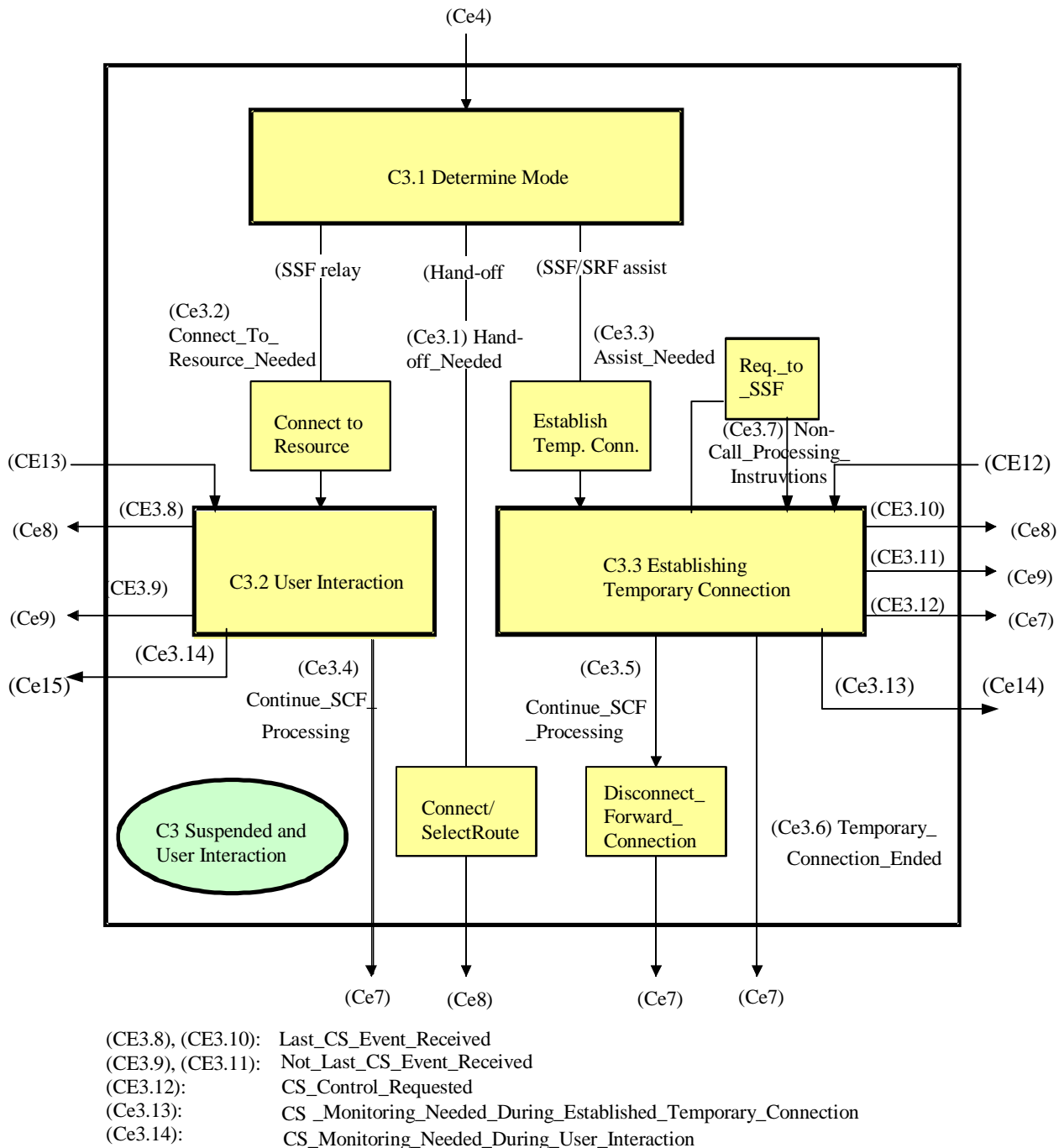
Figure 55: Partial expansion of the state C3 FSM for CS

9.4.3.3.1   State C3.1: "Determine Mode"

The following events are considered in this state:

- (Ce3.1) Hand-off_Needed: this is an internal event that takes place only with the hand-off case. In this case, the SCF sends the **Connect** operation with the handed-off SSF address to the initiating SSF. This event is mapped as the FSM event (Ce8).

- (Ce3.2) Connect_To_Resource_Needed: this is an internal event that takes place only in the case of initiating SSF relay. In this case, the SCF sends the **ConnectToResource** operation to the initiating SSF. This event causes a transition to the state C3.2, **User Interaction**.

- (Ce3.3) Assist_Needed: this is an internal event that takes place when either the assisting SSF or the direct SCF-SRF relation is needed. In this case, the SCF sends the **EstablishTemporaryConnection** operation to the initiating SSF with the assisting SSF address or the assisting SRF address. This event causes a transition to the state C3.3, **Establishing Temporary Connection**.

9.4.3.3.2   State C3.2: "User Interaction"

The following events are considered in this state:

- (Ce3.4) Continue_SCF_Processing: in this case, the SCF has obtained all the information from the SRF, that is needed to instruct the SSF to complete the call. This event is mapped as the FSM event (Ce7).

- (CE3.8) Last_CS_Event_Received: this is an external event caused by a reception of a last event from the corresponding CS. This event causes a transition to the state C1, **CS Control Idle**. This event is mapped as the FSM event (Ce8).

- (CE3.9) Not_Last_CS_Event_Received: this is an external event. This event causes a transition to the state C2.2, **Waiting for Notification or Request**. This event is mapped as the FSM event (Ce9).

- (Ce3.14) CS_Monitoring_Needed_During_User_Interaction: this is an internal event, caused by the service logic because of the need for monitoring during user interaction. This event causes a transition to the state C4.2, **User Interaction Monitoring**. This event is mapped as the FSM event (Ce15).

In order to further describe the detailed procedures relevant to this state, the state is divided into sub-states as illustrated in figure 56.
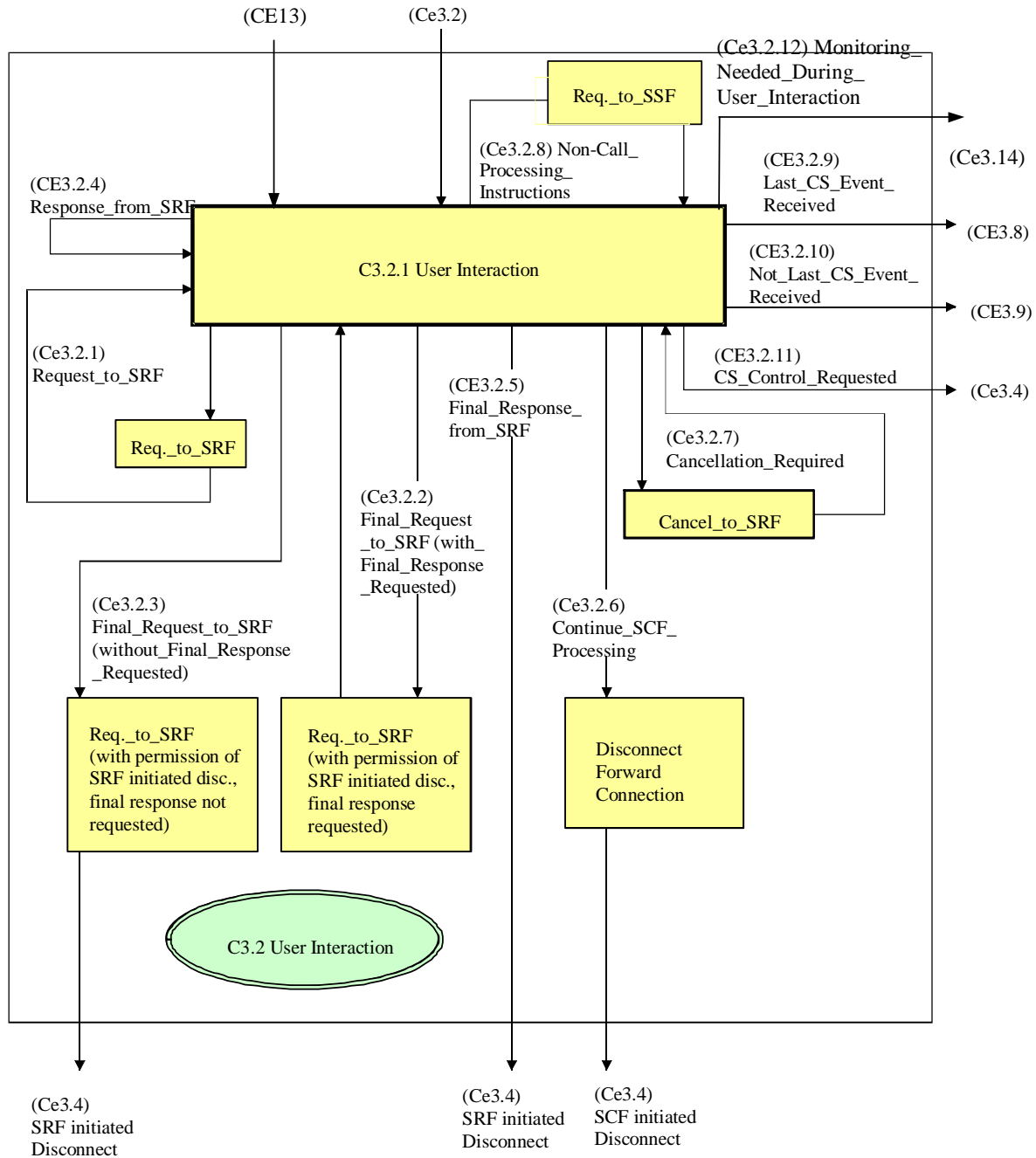
Figure 56: Partial expansion of the state C3.2 FSM for CS

9.4.3.3.2.1   State C3.2.1: "User Interaction"

The following events are considered in this state:

- (Ce3.2.1) Request_to_SRF: this event is an internal event caused by sending one or more of the following operations to the SSF.

    - PlayAnnouncement;

    - PromptAndCollectUserInformation;

    - ScriptRun;

    - ScriptInformation;

    - ScriptClose;

- PromptAndReceiveMessage.

This event causes a transition back to the state C3.2.1, **User Interaction**.

- (Ce3.2.2) Final_Request_to_SRF (with Final Response Requested): this is an internal event that takes place when the FSM for CS instance finishes the user interaction and requests the disconnection of bearer connection between the initiating SSF and the SRF by means of SRF-initiated disconnect. In this case, the SCF sends the **PlayAnnouncement** (containing a request for returning a **SpecializedResourceReport** operation as an indication of completion of the operation) or **PromptAndCollectUserInformation** operation or ScriptRun, ScriptInformation, ScriptClose, or PromptAndReceiveMessage operation with permission of SRF-initiated disconnect to the SRF. In this case, the FSM for CS transits back to the same state.

- (Ce3.2.3) Final_Request_to_SRF (without Final Response Requested): this is an internal event that takes place when the FSM for CS instance finishes the user interaction and requests the disconnection of bearer connection between the initiating SSF and the SRF by means of SRF-initiated disconnect, while no **SpecializedResourceReport** operation has been requested to be returned to the SCF when an announcement is completed. In this case, the SCF sends the **PlayAnnouncement** (not containing a request for returning a **SpecializedResourceReport** operation as an indication of completion of the operation) or **ScriptRun**, **ScriptInformation**, or ScriptClose operation with permission of SRF-initiated disconnect to the SRF. This event is mapped as the FSM event (Ce3.4).

- (CE3.2.4) Response_from_SRF: this is an external event caused by the reception of **SpecializedResourceReport, or ScriptEvent,** or return result from **PromptAndReceiveMessage,** or return result from **PromptAndCollectUserInformation** operation. On the receipt of either, the FSM for CS transits back to the same state.

- (CE3.2.5) Final_Response_from_SRF: this is an external event caused by the reception of **SpecializedResourceReport, or ScriptEvent** or return result from **PromptAndReceiveMessage,** or return result from **PromptAndCollectUserInformation** operation with permission of SRF-initiated disconnect. This event is mapped as the FSM event (Ce3.4).

- (Ce3.2.6) Continue_SCF_Processing: this is an internal event that takes place when the FSM for CS instance finishes the user interaction and requests the disconnection of bearer connection between the initiating SSF and SRF by means of SCF initiated disconnect. In this case, the SCF sends the **DisconnectForwardConnection or DisconnectForwardConnectionWithArgument** operation to the initiating SSF. This event is mapped as the FSM event (Ce3.4).

- (Ce3.2.7) Cancellation_Required: this is an internal event that takes place when the SLPI cancels the previous **PlayAnnouncement** or **PromptAndCollectUserInformation** or PromptAndReceiveMessage operation. In this case, the SCF sends the **Cancel** operation to the SSF. The FSM for CS transits back to the same state.

- (Ce3.2.8) Non-Call_Processing_Instructions: this is an internal event caused by the service logic when there is a need to send one or more of the following operations to the SSF:

  - ApplyCharging;

  - FurnishChargingInformation;

  - RequestNotificationChargingEvent;

  - SendChargingInformation;

  - SetServiceProfile.

When the application timer $T_{SCF-SSF}$ expires, the FSM for CS sends a ResetTimer operation to the SSF for the corresponding CS.

The FSM for CS transits back to the same state.

- (CE3.2.9) Last_CS_Event_Received: this is an external event caused by a reception of one of the following operations from the SSF:

  - CallInformationReport;

  - ApplyChargingReport;

- EventReportBCSM **(for Abandon/Disconnect EDP-N);**

- EntityReleased.

In this case, there is neither an outstanding armed EDP, a pending **CallInformationReport,** nor a pending **ApplyChargingReport** operation. This event causes a transition to the state C1, **CS Control Idle**. This event is mapped as the FSM event (CE3.8).

- (CE3.2.10) Not_Last_CS_Event_Received: this is an external event caused by a reception of one of the following operations from the SSF:

  - CallInformationReport;

  - ApplyChargingReport;

  - EventReportBCSM **(for Abandon/Disconnect EDP-N).**

In this case, there is still an outstanding armed EDP or pending **CallInformationReport** or **ApplyChargingReport** operation. This event causes a transition to the state C2.2, **Waiting for Notification or Request**. This event is mapped as the FSM event (CE3.9).

- (CE3.2.11) CS_Control_Requested: this is an external event caused by a reception of one of the following operations from the SSF:

  - EventReportBCSM (for Abandon/Disconnect EDP-R).

This event causes a transition to the state C2.1, **Preparing CS Instructions**. This event is mapped as the FSM event (Ce3.4).

- (Ce3.2.12) Monitoring_Needed_During_User_Interaction: this is an internal event caused by the service logic when there is a need to bring the SSF in the processing state, by sending one of the following operations to the SSF:

  - Continue;

  - ContinueWithArgument.

This event causes a transition to state C4.2, **User Interaction Monitoring**. This event is mapped as the FSM event (Ce3.14).

It should be noted that the bearer connection between the SSF and the SRF is disconnected when the FSM for CS exits from this state except the (Ce3.2.12) event.

9.4.3.3.3   State C3.3: "Establishing Temporary Connection"

The following events are considered in this state:

- (Ce3.5) Continue_SCF_Processing: this is an internal event that takes place when the FSM instance for CS finishes the user interaction and requests the disconnection of bearer connection between the initiating SSF and the assisting SSF/SRF by means of SCF initiated disconnect. In this case, the SCF sends the **DisconnectForwardConnection** or **DisconnectForwardConnectionWithArgument** operation to the initiating SSF. This event is mapped as the FSM event (Ce7).

- (Ce3.6) Temporary_Connection_Ended: this is an internal event caused by the notification from the FSM instance for CSA because of the end of the user interaction for the assisting SRF. This event is mapped as the FSM event (Ce7).

- (Ce3.7) Non-Call_Processing_Instructions: this is an internal event caused by the service logic when there is a need to send such an operation to the SSF. It causes one or more of the following operations to be issued to the SSF:

  - ApplyCharging;

  - FurnishChargingInformation;

- RequestNotificationChargingEvent;

- SendChargingInformation;

- SetServiceProfile.

When the application timer $T_{SCF-SSF}$ expires, the FSM for CS sends a ResetTimer operation to the SSF for the corresponding CS.

The FSM for CS transits back to the same state.

- (CE3.10) Last_CS_Event_Received: this is an external event caused by a reception of one of the following operations from the SSF:

  - CallInformationReport;

  - ApplyChargingReport;

  - EventReportBCSM **(for Abandon/Disconnect EDP-N);**

  - EntityReleased.

In this case, there is neither an outstanding armed EDP, a pending **CallInformationReport,** nor a pending **ApplyChargingReport** operation. This event causes a transition to the state C1, **CS Control Idle**. This event is mapped as the FSM event (Ce8).

- (CE3.11) Not_Last_CS_Event_Received: this is an external event caused by a reception of one of the following operations from the SSF:

  - CallInformationReport;

  - ApplyChargingReport;

  - EventReportBCSM **(for Abandon/Disconnect EDP-N)**.

In this case, there is still an outstanding armed EDP or pending **CallInformationReport** or **ApplyChargingReport** operation. This event causes a transition to the state C2.2, **Waiting for Notification or Request**. This event is mapped as the FSM event (Ce9).

- (CE3.12) CS_Control_Requested: this is an external event caused by a reception of one of the following operations from the SSF:

  - EventReportBCSM (for Abandon/Disconnect EDP-R).

This event causes a transition to the state C2.1, **Preparing CS Instructions**. This event is mapped as the FSM event (Ce7).

- (Ce3.13) CS_Monitoring_Needed_During_Established_Temporary_Connection: this is an internal event, caused by the service logic when there is a need for monitoring, while being in state C3.3, Establishing Temporary Connection. In this case one of the following operations is sent to the SSF:

  - Continue;

  - ContinueWithArgument.

This event causes a transition to the state C4.3, **Establishing Temporary Connection Monitoring**. This event is mapped as the FSM event (Ce14).

9.4.3.4   State C4: "Not Suspended and User Interaction"

The following events are considered in this state:

- (Ce7) Continue_SCF_Processing: in this case, the SCF has obtained all the information from the SRF, that is needed to instruct the SSF to complete the call. This event causes a transition to state C2.1, **Preparing CS instructions**.

- (Ce8) Processing_Completed: this is an internal event caused by the end of processing for the CS. This event causes a transition to the state C1, **CS Control Idle**.

- (Ce9) CS_Monitoring_Needed: this is an internal event caused by the necessity of monitoring. This event causes a transition to the state C2.2, **Waiting for Notification or Request**.

- (CE16) MidCall_EDP_N: this event is an external event caused by a reception of one of the following operations if O/T-MidCall EDP-N was armed to be reported in "any state":

    - **EventReportBCSM** (for O/T-MidCall).

This event causes a transition back to the same state.

- (CE12) CS_Instruction_Needed_During_Establish_Temporary_Connection: this is an external event caused by an EventReportBCSM operation for EDP-R while being in state C4.3 Establishing Temporary Connection Monitoring. This event causes a transition to the state C3.3, Establishing Temporary Connection.

- (CE13) CS_Instruction_Needed_During_User_Interaction: this is an external event caused by an EventReportBCSM operation for EDP-R while being in the state C4.2 User Interaction Monitoring. This event causes a transition to the state C3.2, User Interaction.

On order to further describe the procedures relevant to this state, the state is divided into sub-states as illustrated in figure 57.
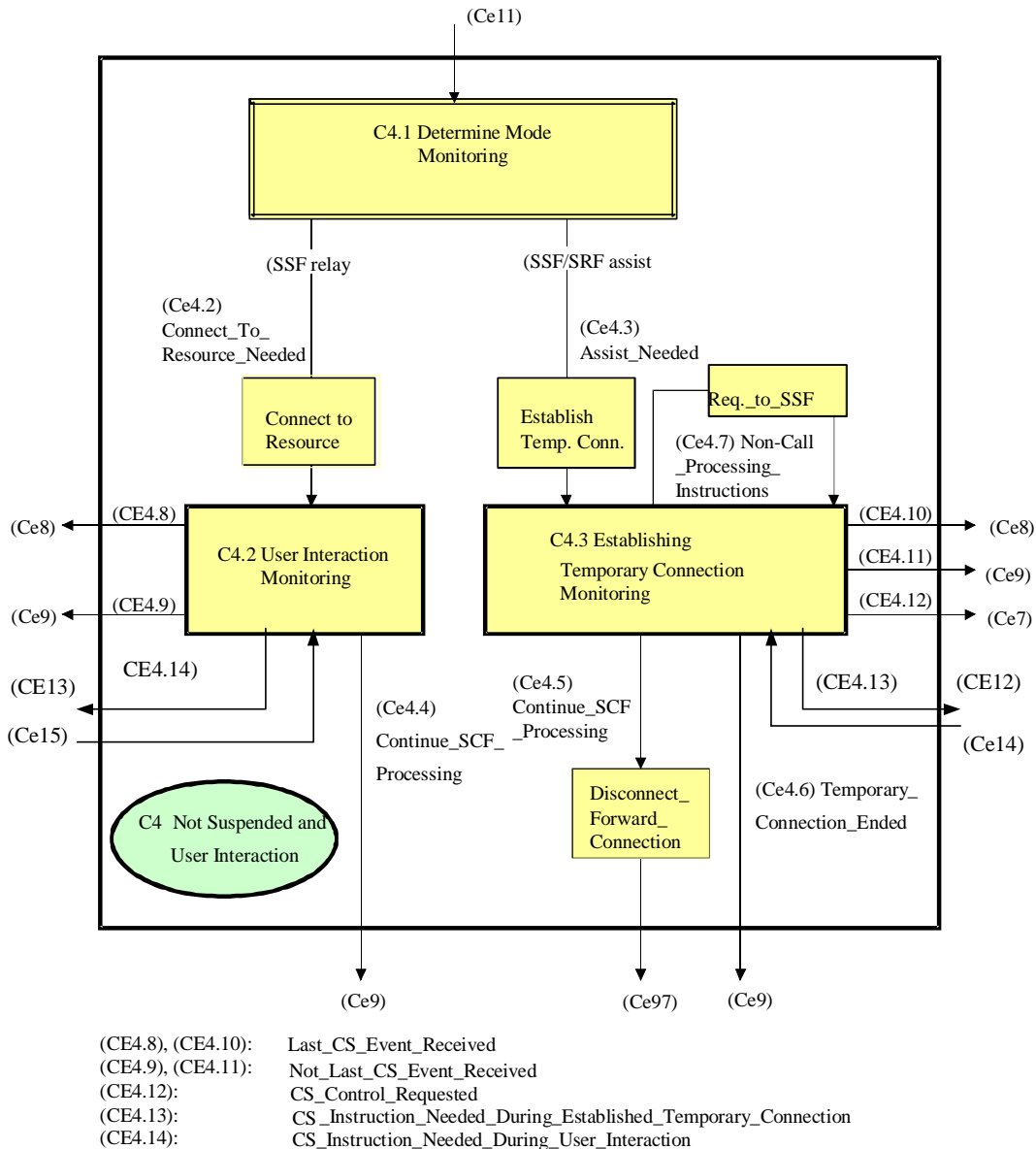
Figure 57: Partial expansion of the state C4 FSM for CS

9.4.3.4.1   State C4.1: "Determine Mode Monitoring"

The following events are considered in this state:

- (Ce4.2) Connect_To_Resource_Needed: this is an internal event that takes place only in the case of initiating SSF relay. In this case, the SCF sends the **ConnectToResource** operation to the initiating SSF. This event causes a transition to the state C4.2, **User Interaction_Monitoring**.

- (Ce4.3) Assist_Needed: this is an internal event that takes place when either the assisting SSF or the direct SCF-SRF relation is needed. In this case, the SCF sends the **EstablishTemporaryConnection** operation to the initiating SSF with the assisting SSF address or the assisting SRF address. This event causes a transition to the state C4.3, **Establishing Temporary Connection Monitoring**.

9.4.3.4.2   State C4.2: "User Interaction Monitoring"

The following events are considered in this state:

- (Ce4.4) Continue_SCF_Processing: in this case, the SCF has obtained all the information from the SRF, that is needed to instruct the SSF to complete the call. This event is mapped as the FSM event (Ce9).

- (CE4.8) Last_CS_Event_Received: this is an external event caused by a reception of a last event from the corresponding CS. This event causes a transition to the state C1, **CS Control Idle**. This event is mapped as the FSM event (Ce8).

- (CE4.9) Not_Last_CS_Event_Received: this is an external event. This event causes a transition to the state C2.2, **Waiting for Notification or Request**. This event is mapped as the FSM event (Ce9).

- (CE4.14) CS_Instruction_Needed_During_User_Interaction: this is an external event. This event causes a transition to the state C3.2, **User Interaction**. This event is mapped as the FSM event (CE13).

In order to further describe the detailed procedures relevant to this state, the state is divided into sub-states as illustrated in figure 58.
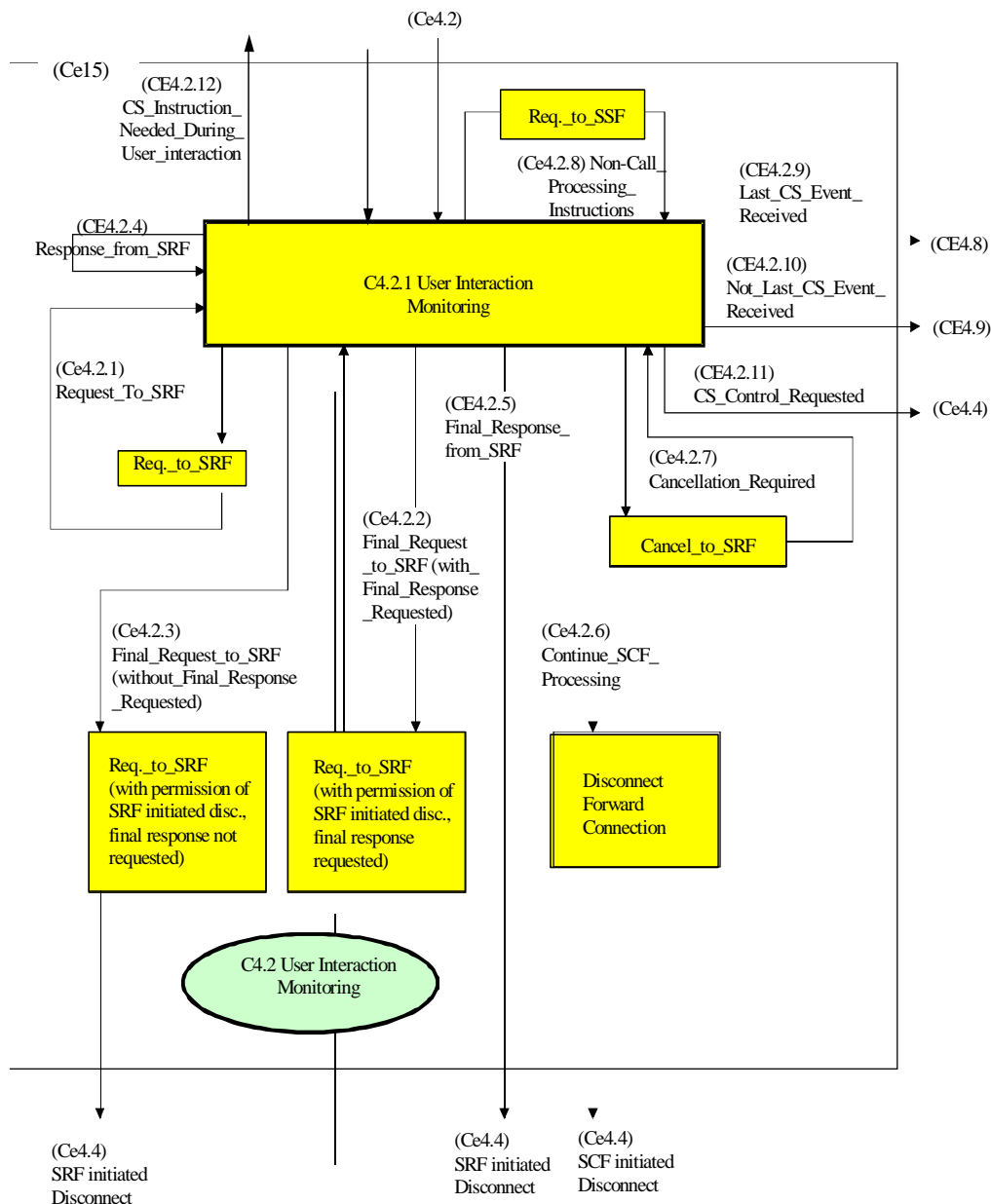
Figure 58: Partial expansion of the state C4.2 FSM for CS

9.4.3.4.2.1   State C4.2.1: "User Interaction Monitoring"

The following events are considered in this state:

- (Ce4.2.1) Request_to_SRF: this event is an internal event caused by sending one or more of the following operations to the SSF.

  - PlayAnnouncement.

This event causes a transition back to the state C4.2.1, **User Interaction Monitoring**.

- (Ce4.2.2) Final_Request_to_SRF (with Final Response Requested): this is an internal event that takes place when the FSM for CS instance finishes the user interaction and requests the disconnection of bearer connection between the initiating SSF and the SRF by means of SRF-initiated disconnect. In this case, the SCF sends the **PlayAnnouncement** (containing a request for returning a **SpecializedResourceReport** operation as an indication of completion of the operation) In this case, the FSM for CS transits back to the same state.

- (Ce4.2.3) Final_Request_to_SRF (without Final Response Requested): this is an internal event that takes place when the FSM for CS instance finishes the user interaction and requests the disconnection of bearer connection between the initiating SSF and the SRF by means of SRF-initiated disconnect, while no **SpecializedResourceReport** operation has been requested to be returned to the SCF when an announcement is completed. In this case, the SCF sends the **PlayAnnouncement** (not containing a request for returning a **SpecializedResourceReport** operation as an indication of completion of the operation) with permission of SRF-initiated disconnect to the SRF. This event is mapped as the FSM event (Ce4.4).

- (CE4.2.4) Response_from_SRF: this is an external event caused by the reception of **SpecializedResourceReport**. On the receipt, the FSM for CS transits back to the same state.

- (CE4.2.5) Final_Response_from_SRF: this is an external event caused by the reception of **SpecializedResourceReport** with permission of SRF-initiated disconnect. This event is mapped as the FSM event (Ce4.4).

- (Ce4.2.6) Continue_SCF_Processing: this is an internal event that takes place when the FSM for CS instance finishes the user interaction and requests the of bearer connection between the initiating SSF and SRF by means of SCF disconnection initiated disconnect. In this case, the SCF sends the **DisconnectForwardConnection** or **DisconnectForwardConnectionWithArgument** operation to the initiating SSF. This event is mapped as the FSM event (Ce4.4).

- (Ce4.2.7) Cancellation_Required: this is an internal event that takes place when the SLPI cancels the previous **PlayAnnouncement** operation. In this case, the SCF sends the **Cancel** operation to the SSF. The FSM for CS transits back to the same state.

- (Ce4.2.8) Non-Call_Processing_Instructions: this is an internal event caused by the service logic when there is a need to send one or more of the following operations to the SSF:

  - ApplyCharging;

  - FurnishChargingInformation;

  - RequestNotificationChargingEvent; and

  - SendChargingInformation;

  - SetServiceProfile.

When the application timer $T_{SCF-SSF}$ expires, the FSM for CS sends a ResetTimer operation to the SSF for the corresponding CS.

The FSM for CS transits back to the same state.

- (CE4.2.9) Last_CS_Event_Received: this is an external event caused by a reception of one of the following operations from the SSF:

  - CallInformationReport;

- ApplyChargingReport;

- EventReportBCSM **(for Abandon/Disconnect EDP-N);**

- EntityReleased**.**

In this case, there is neither an outstanding armed EDP, a pending **CallInformationReport,** nor a pending **ApplyChargingReport** operation. This event causes a transition to the state C1, **CS Control Idle**. This event is mapped as the FSM event (CE4.8).

- (CE4.2.10) Not_Last_CS_Event_Received: this is an external event caused by a reception of one of the following operations from the SSF:

- CallInformationReport;

- ApplyChargingReport;

- EventReportBCSM **(for Abandon/Disconnect EDP-N).**

In this case, there is still an outstanding armed EDP or pending **CallInformationReport** or **ApplyChargingReport** operation. This event causes a transition to the state C2.2, **Waiting for Notification or Request**. This event is mapped as the FSM event (CE4.9).

- (CE4.2.11) CS_Control_Requested: this is an external event caused by a reception of one of the following operations from the SSF:

- EventReportBCSM (for Abandon/Disconnect EDP-R).

This event causes a transition to the state C2.1, **Preparing CS Instructions**. This event is mapped as the FSM event (Ce4.4).

- (CE4.2.12) CS_Instruction _Needed_During_User_Interaction: this is an external event caused by the reception of one of the following operations which notifies the SCF of EDP-R except Abandon and Disconnect:

- EventReportBCSM **(EDP-R)**; and

this event causes a transition to state C3.2, **User Interaction**. This event is mapped as the FSM event (CE4.14).

9.4.3.4.3   State C4.3: "Establishing Temporary Connection Monitoring"

NOTE: ETC or CTR in the transparent relay case is accepted in the Monitoring state, but it is up to the SCF to ensure that the User Interaction through the assisting SRF will be limited to PlayAnnouncement.

The following events are considered in this state:

- (Ce4.5) Continue_SCF_Processing: this is an internal event that takes place when the FSM instance for CS finishes the user interaction and requests the disconnection of bearer connection between the initiating SSF and the assisting SSF/SRF by means of SCF initiated disconnect. In this case, the SCF sends the **DisconnectForwardConnection** or **DisconnectForwardConnectionWithArgument** operation to the initiating SSF. This event is mapped as the FSM event (Ce7).

- (Ce4.6) Temporary_Connection_Ended: this is an internal event caused by the notification from the FSM instance for CSA because of the end of the user interaction for the assisting SRF. This event is mapped as the FSM event (Ce9).

- (Ce4.7) Non-Call_Processing_Instructions: this is an internal event caused by the service logic when there is a need to send such an operation to the SSF. It causes one or more of the following operations to be issued to the SSF:

- ApplyCharging;

- FurnishChargingInformation;

- RequestNotificationChargingEvent; and

- SendChargingInformation;

- SetServiceProfile.

When the application timer $T_{SCF-SSF}$ expires, the FSM for CS sends a ResetTimer operation to the SSF for the corresponding CS.

The FSM for CS transits back to the same state.

- (CE4.10) Last_CS_Event_Received: this is an external event caused by a reception of one of the following operations from the SSF:

    - CallInformationReport;

    - ApplyChargingReport;

    - EventReportBCSM **(for Abandon/Disconnect EDP-N);**

    - EntityReleased**.**

In this case, there is neither an outstanding armed EDP, a pending **CallInformationReport,** nor a pending **ApplyChargingReport** operation. This event causes a transition to the state C1, **CS Control Idle**. This event is mapped as the FSM event (Ce8).

- (CE4.11) Not_Last_CS_Event_Received: this is an external event caused by a reception of one of the following operations from the SSF:

    - CallInformationReport;

    - ApplyChargingReport;

    - EventReportBCSM **(for Abandon/Disconnect EDP-N).**

In this case, there is still an outstanding armed EDP or pending **CallInformationReport** or **ApplyChargingReport** operation. This event causes a transition to the state C2.2, **Waiting for Notification or Request**. This event is mapped as the FSM event (Ce9).

- (CE4.12) CS_Control_Requested: this is an external event caused by a reception of one of the following operations from the SSF:

- EventReportBCSM (for Abandon/Disconnect EDP-R).

This event causes a transition to the state C2.1, **Preparing CS Instructions**. This event is mapped as the FSM event (Ce7).

- (CE4.13) CS_Instruction_Needed_During_Established_Temporary_Connection: this is an external event, caused by a reception of one of the following operations which notifies the SCF of EDP-R except Abandon and Disconnect:

    - EventReportBCSM (EDP-R).

This event causes a transition to the state C3.3, **Establishing Temporary Connection**. This event is mapped as the FSM event (CE12).

9.4.4   Finite State Model for Assisting SSF

Figure 59 shows the general State Diagram of the FSM for assisting SSF as relevant to the procedures concerning the SCF FSM part of the SCP/AD/SN during the processing of an IN call.
Each state is discussed in one of the following clauses.

The FSM for assisting SSF has an application timer, $T_{SCF-SSF}$, whose purpose is to restart the timer, T_{SSF}, to guard the association between the assisting SSF and the SCF.
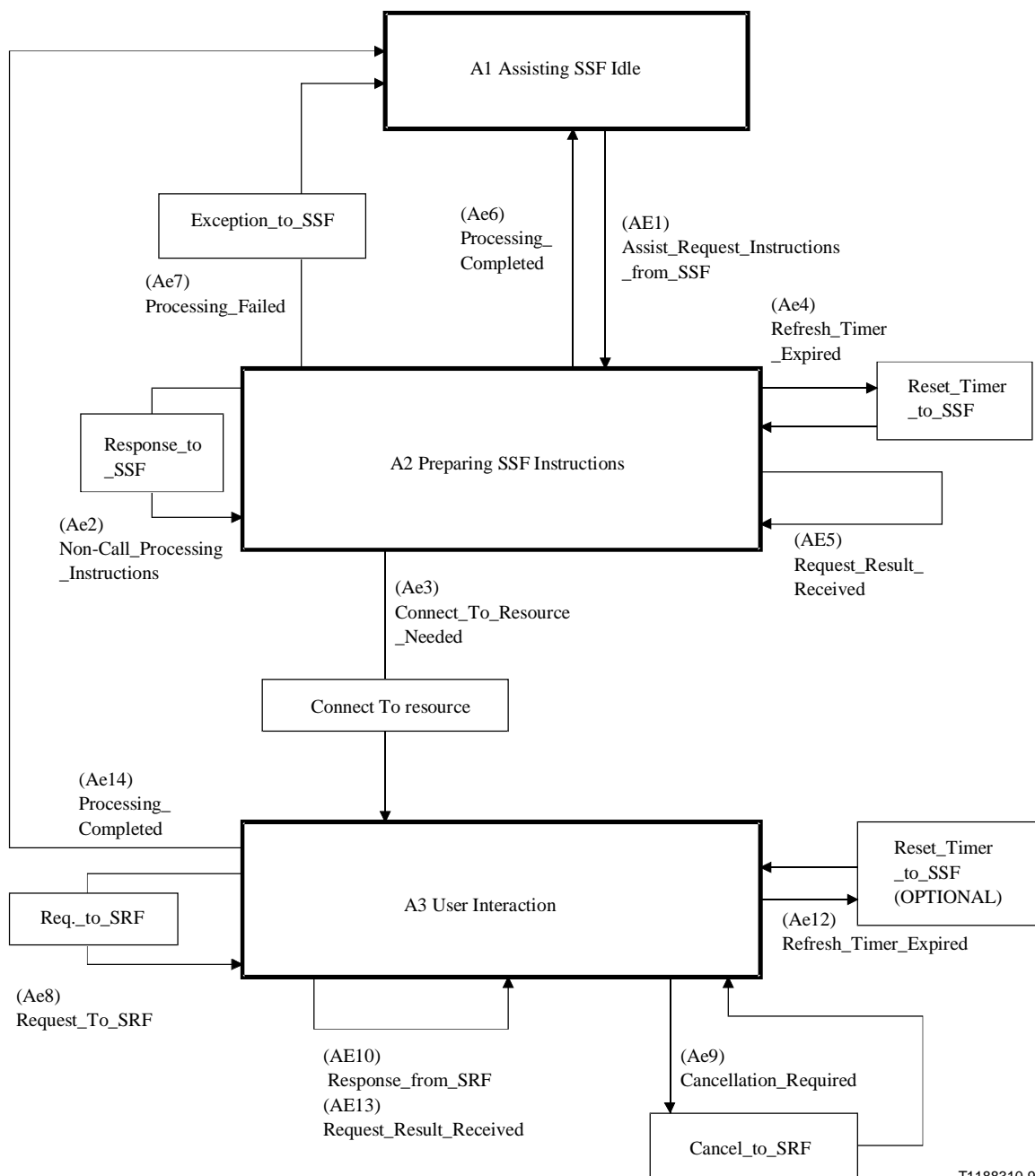
Figure 59: FSM for Assisting SSF

Timer $T_{SCF-SSF}$ is set in the following cases:

- when the SCF receives an **AssistRequestInstructions** operation. In this case, this timer is restart when a first request, other than **ResetTimer** operation, is sent to the assisting SSF. On the expiration of timer $T_{SCF-SSF}$, the **FSM for assisting SSF** may restart T_{SSF} once using the **ResetTimer** operation, and restart timer $T_{SCF-SSF}$. On the second expiration of $T_{SCF-SSF}$, the **FSM for assisting SSF** informs the SLPI and the maintenance functions, and the **FSM for assisting SSF** transitions to the state A1, Assisting SSF Idle;

- when **FSM for assisting SSF** enters the "User Interaction" state. In this case, on the expiration of $T_{SCF-SSF}$, the SCF may restart T_{SSF} using the **ResetTimer** operation any number of times (OPTIONAL).

NOTE: The word "OPTIONAL" refers to the use of the application timer $T_{SCF-SSF}$. Whether it is used depends on an implementation, but, if used, it must be synchronized with T_{SSF} in the assisting SSF FSM.

In all two cases, $T_{SCF-SSF}$ may respectively have two different values as defined by the application. The values of $T_{SCF-SSF}$ are smaller than the respective values of T_{SSF}.

When receiving or sending any other operation, the SCF should restart $T_{SCF-SSF}$.

9.4.4.1   State A1: "Assisting SSF Idle"

The following events are considered in this state:

ï (AE1) Assist_Request_Instructions_from_SSF: This is an external event caused by a reception of the following operation:

   - **AssistRequestInstructions.**

This event causes a transition to the state A2, **Preparing SSF Instructions**.

9.4.4.2   State A2: "Preparing SSF Instructions"

The following events are considered in this state:

ï (Ae2) Non-Call_Processing_Instructions: This is an internal event caused by the SLPI when there is a need to send such an operation to the assisting SSF. It causes one or more of the following operations to be issued to the SSF:

   - **ApplyCharging;**

   - **FurnishChargingInformation;**

   - **SendChargingInformation.**

This event causes a transition back to the state A2, **Preparing SSF Instructions**.

ï (Ae3) Connect_To_Resource_Needed: this is an internal event. In this case, the SCF sends the **ConnectToResource** operation to the assisting SSF. This event causes a transition to the state A3, **User Interaction**.

ï (Ae4) Refresh_Timer_Expired: This is an internal event, which results in sending the **ResetTimer** operation to the assisting SSF and a transition back to the same state.

ï (AE5) Request_Result_Received: This is an external event caused by a reception of the following operations from the assisting SSF:

   - **ApplyChargingReport.**

This event causes a transition to the same state.

ï (Ae6) Processing_Completed: This is an internal event, caused by the end of processing for the assisting SSF. This event causes a transition to the state A1, **Assisting SSF Idle**.

ï (Ae7) Processing_Failed: This (internal) event causes an appropriate exception processing and a transition back to the state A1, **Assisting SSF Idle**.

9.4.4.3   State A3: "User Interaction"

The following events are considered in this state:

ï (Ae8) Request_To_SRF: this event is a internal event caused by sending one or more of the following operations to the assisting SSF:

- **PlayAnnouncement;**

- **PromptAndCollectUserInformation;**

- **ScriptRun;**

- **ScriptInfo;**

- **ScriptClose;**

- **PromptAndReceiveMessage.**

This event causes a transition back to the state A3, **User Interaction**.

ï (Ae9) Cancellation_Required: this is an internal event that takes place when the SLPI cancels the previous **PlayAnnouncement** or **PromptAndCollectUserInformation** operation. In this case, the SCF sends the **Cancel** operation to the assisting SSF. The **FSM for assisting SSF** transits back to the same state.

ï (AE10) Response_from_SRF: this is an external event caused by the reception of **SpecializedResourceReport** or return result from **PromptAndCollectUserInformation** operation or PromptAndReceiveMessage Result or ScriptEvent. On the receipt of either, the FSM for assisting SSF transits back to the same state.

ï (Ae12) Refresh_Timer_Expired: This is an internal event, which results in sending the **ResetTimer** operation to the assisting SSF and a transition back to the same state.

ï (AE13) Request_Result_Received: This is an external event caused by a reception of the following operation from the assisting SSF:

- **ApplyChargingReport.**

This event causes a transition to the same state.

ï (Ae14) Processing_Completed: This is an internal event, caused by the end of processing for the assisting SSF. This event causes a transition to the state A1, **Assisting SSF Idle**.

9.4.5   Finite State Model for Handed-off SSF

Figure 60 shows the general State Diagram of the **FSM for handed-off SSF** as relevant to the procedures concerning the SCF FSM part of the SCP/AD/SN during the processing of an IN call. Each state is discussed in one of the following clauses. The Hand-Off FSM applies only to the case where final treatment is to be applied.
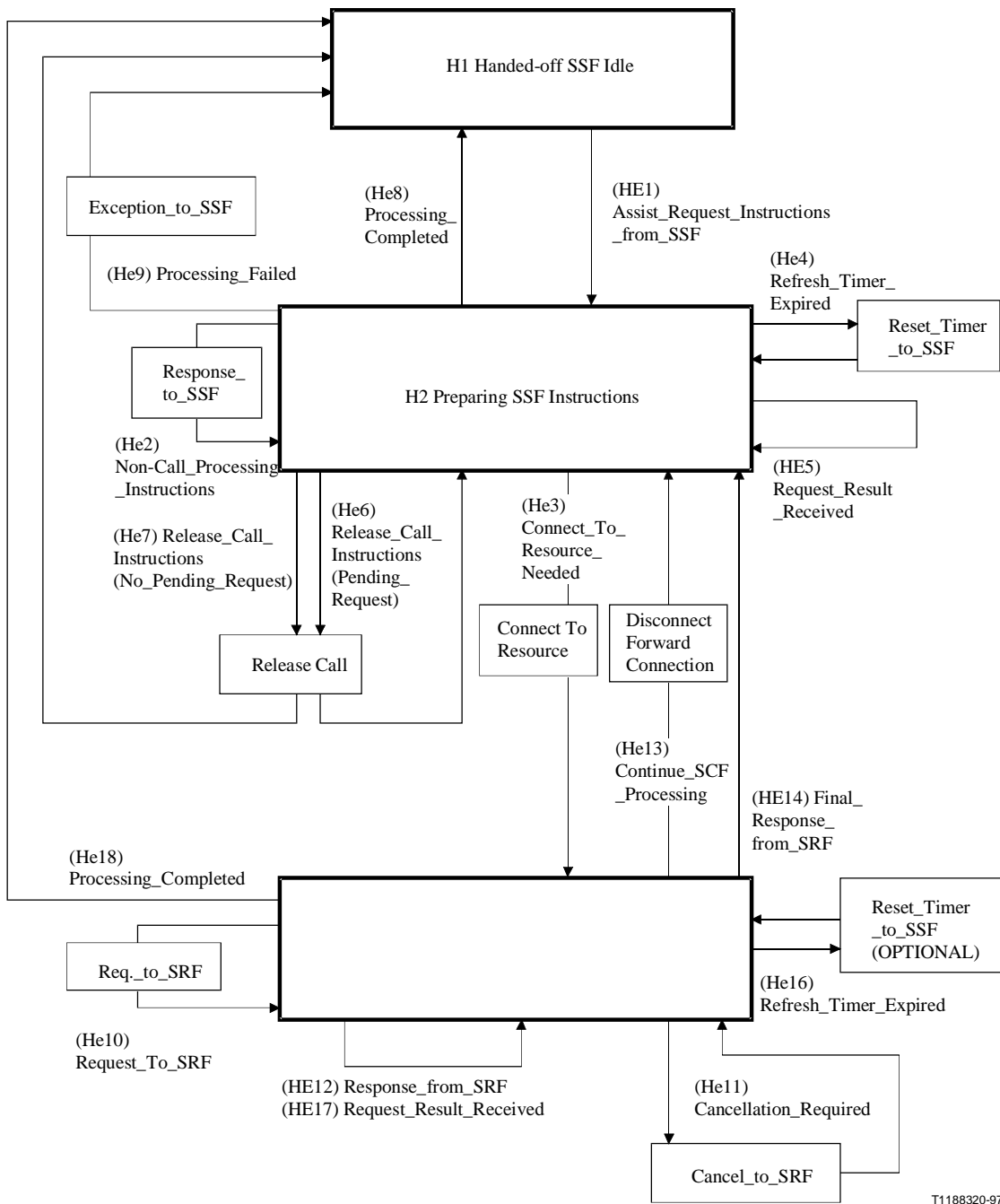
Figure 60: FSM for Handed-off SSF

The **FSM for handed-off SSF** has an application timer, $T_{SCF\text{-}SSF}$, whose purpose is to restart the timer, T_{SSF}, to guard the association between the handed-off SSF and the SCF.

Timer $T_{SCF-SSF}$ is set in the following cases:

- when the SCF receives an **AssistRequestInstructions** operation. In this case, this timer is restart when a first request, other than **ResetTimer** operation, is sent to the handed-off SSF. On the expiration of timer $T_{SCF-SSF}$, the **FSM for handed-off SSF** may restart T_{SSF} once using the **ResetTimer** operation, and restart timer $T_{SCF-SSF}$. On the second expiration of $T_{SCF-SSF}$, the **FSM for handed-off SSF** informs the SLPI and the maintenance functions, and the **FSM for handed-off SSF** transitions to the state H1, **Handed-off SSF Idle**;

- when **FSM for handed-off SSF** enters the "**User Interaction**" state. In this case, on the expiration of $T_{SCF-SSF}$, the SCF may restart T_{SSF} using the **ResetTimer** operation any number of times (OPTIONAL).

NOTE: The word "OPTIONAL" refers to the use of the application timer $T_{SCF-SSF}$. Whether it is used depends on an implementation, but, if used, it must be synchronized with T_{SSF} in the handed-off SSF FSM.

In all two cases, $T_{SCF-SSF}$ may respectively have two different values as defined by the application. The values of $T_{SCF-SSF}$ are smaller than the respective values of T_{SSF}.

When receiving or sending any other operation, the SCF should restart $T_{SCF-SSF}$.

9.4.5.1   State H1: "Handed-off SSF Idle"

The following events are considered in this state:

(HE1) Assist_Request_Instructions_from_SSF: This is an external event caused by a reception of the following operation:

- **AssistRequestInstructions.**

This event causes a transition to the state H2, **Preparing SSF Instructions**.

9.4.5.2   State H2: "Preparing SSF Instructions"

The following events are considered in this state:

(He2) Non-Call_Processing_Instructions: This is an internal event caused by the SLPI when there is a need to send such an operation to the handed-off SSF. It causes one or more of the following operations to be issued to the SSF:

- **ApplyCharging;**

- **FurnishChargingInformation;**

- **SendChargingInformation.**

This event causes a transition back to the state H2, **Preparing SSF Instructions**.

(He3) Connect_To_Resource_Needed: this is an internal event. In this case, the SCF sends the **ConnectToResource** operation to the handed-off SSF. This event causes a transition to the state H3, **User Interaction**.

(He4) Refresh_Timer_Expired: This is an internal event, which results in sending the **ResetTimer** operation to the handed-off SSF and a transition back to the same state.

(HE5) Request_Result_Received: This is an external event caused by a reception of the following operations from the handed-off SSF:

- **ApplyChargingReport.**

This event causes a transition to the same state.

(He6) Release_Call_Instructions (Pending_Request): This is an internal event caused by sending a ReleaseCall operation when there are one or more pending requests (CallInformationRequest or ApplyCharging). This event causes a transition to the same.

(He7) Release_Call_Instructions (No_Pending_Request): This is an internal event caused by sending a ReleaseCall operation when there are no pending requests (neither CallInformationRequest nor ApplyCharging). This event causes a transition to the state H1, Handed-off SSF Idle.

(He8) Processing_Completed: This is an internal event, caused by the end of processing for the handed-off SSF. This event causes a transition to the state H1, **Handed-off SSF Idle**.

(He9) Processing_Failed: This (internal) event causes an appropriate exception processing and a transition back to the state H1, **Handed-off SSF Idle**.

9.4.5.3   State H3: "User Interaction"

The following events are considered in this state:

(He10) Request_To_SRF: this event is a internal event caused by sending one or more of the following operations to the handed-off SSF.

- **PlayAnnouncement**; and

- **PromptAndCollectUserInformation**.

    - ScriptRun;

    - ScriptInfo;

    - ScriptClose;

    - PromptAndReceiveMessage.

This event causes a transition back to the state H3, **User Interaction**.

(He11) Cancellation_Required: this is an internal event that takes place when the SLPI cancels the previous **PlayAnnouncement** or **PromptAndCollectUserInformation** operation. In this case, the SCF sends the **Cancel** operation to the handed-off SSF. The **FSM for handed-off SSF** transits back to the same state.

(HE12) Response_from_SRF: this is an external event caused by the reception of **SpecializedResourceReport** or return result from **PromptAndCollectUserInformation** operation or PromptAndReceiveMessage Result or ScriptEvent. On the receipt of either, the FSM for handed-off SSF transits back to the same state.

(He13) Continue_SCF_Processing: this is an internal event that takes place when the **FSM for handed-off SSF** instance finishes the user interaction and requests the disconnection of bearer connection between the handed-off SSF and SRF by means of SCF initiated disconnect. In this case, the SCF sends the **DisconnectForwardConnection** or **DisconnectForwardConnectionwithArgument** operation to the handed-off SSF. The **FSM for handed-off SSF** transits to the state H2, **Preparing SSF Instructions**.

(HE14) Final_Response_from_SRF: this is an external event caused by the reception of **SpecializedResourceReport** or return result from **PromptAndCollectUserInformation** operation or PromptAndReceiveMessage Result or ScriptEvent with the permission of SRF-initiated disconnect. The **FSM for handed-off SSF** transits to the state H2, **Preparing SSF Instructions**.

*(He16) Refresh_Timer_Expired: This is an internal event, which results in sending the **ResetTimer** operation to the handed-off SSF and a transition back to the same state.*

(HE17) Request_Result_Received: This is an external event caused by a reception of the following operations from the handed-off SSF:

- **ApplyChargingReport.**

This event causes a transition to the same state.

(He18) Processing_Completed: This is an internal event, caused by the end of processing for the handed-off SSF. This event causes a transition to the state H1, **Handed-off SSF Idle**.

10   FSM for USI

10.1   USI FSM for SSF

The **SSF_USI FSM** illustrates state transitions for requesting and cancelling the monitoring of the reception of UTSI operations and the sending of STUI operations. This FSM has two states which are "Monitoring USI" and "Idle".
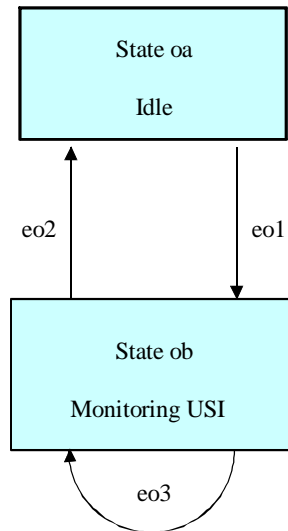
Figure 61: FSM for SSF_USI

The **SSF_USI FSM** transitions are defined in the following way:

eo1   The SCF requests the SSF to monitor the receipt of an USI data for a given *USIServiceIndicator* value by sending a RequestReportUTSI operation for a particular party indicated by the leg identifier by setting the "USIMonitorMode" value to "monitoringActive".

eo2   The SCF requests the SSF to stop monitoring the reception of an USI data for a given *USIServiceIndicator* value by sending a RequestReportUTSI operation for a particular party indicated by the leg identifier by setting the "USIMonitorMode" value to "monitoringInactive".

eo3   The SCF either sends an USI data by means of a SendSTUI operation to the user indicated by the leg identifier for a given *USIServiceIndicator* value; or receives a USI data by means of a ReportUTSI operation.

NOTE 1:   As an SCF controls or monitors the call, the **FSM for CS** is in any state except "Idle"; but the OCCRUI mechanism does not cause any transition in the **FSM for CS**. When the **FSM for CS** returns to the Idle state then also the **FSM for SSF_USI** returns to Idle. SCF may send the operation SendSTUI in the state Idle of the **FSM for SSF_USI.**

NOTE 2:   Refer to EN 301 931-1 for a definition of the term "User" in the context of the OCCRUI mechanism. A *User to Service Information (USI) data* transfer refers to either a User To Service Information (UTSI) or a Service To User Information (STUI) data transfer request.

10.2   USI FSM for SCF

The **USI_SCF FSM** illustrates the SCF monitoring or not of the receipt of the USI data. This FSM has two states which are "Monitoring USI information" and "Idle".
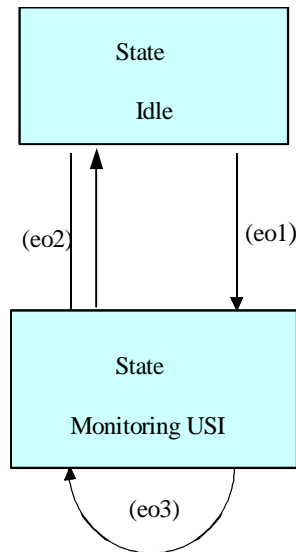
Figure 62: USI_SCF FSM

The **USI_SCF FSM** transitions are defined in the following way:

(eo1):   The SCF requests the SSF to monitor the receipt of an USI data with a given *USIServiceIndicator* value.

(eo2):   The SCF is no longer interested in the receipt of an USI data with the given *USIServiceIndicator* value.

(eo3):   The SCF sends a SendSTUI operation with USI data to the User and/or receives an USI data from the User with the given *USIServiceIndicator* value.

With the same operation, the SCF requests the SSF to monitor or to stop monitoring the receipt of an USI data with a given *USIServiceIndicator* value.

NOTE 1:   As an SCF controls or monitors the call, the **FSM for CS** is in any state except "Idle"; but the OCCRUI mechanism does not cause any transition in the **FSM for CS**.

NOTE 2:   Refer to EN 301 931-1 for a definition of the term "User" in the context of the OCCRUI mechanism. A *User to Service Information (USI) data* transfer refers to either a User To Service Information (UTSI) or a Service To User Information (STUI) data transfer request.

11   Operation Procedures

In case of discrepancies between the Call Segment Connection View (CSCV) state transition tables and CSCV state diagrams described in clause 6 and the state transition tables provided in the following detailed operation procedure descriptions, the latter tables take precedence.

11.1   ActivateServiceFiltering procedure

11.1.1   General description

When receiving this operation, the SSF handles calls to destinations in a specified manner without request for instructions to the SCF. In the case of service filtering the SSF executes a specific service-filtering algorithm. For the transfer of service filtering results refer to the operation "ServiceFilteringResponse".

The ActivateServiceFiltering operation can be received outside a call context transaction only.

11.1.1.1   Parameters

11.1.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- filteredCallTreatment;
  This parameter comprises the following subparameters:

  - sFBillingChargingCharacteristics;

  - informationToSend;

  - maximumNumberOfCounters;

  - releaseCause;

  - sFTariffMessage.

- filteringCharacteristics;
  This parameter comprises the following alternatives:

  - interval; or

  - numberOfCalls.

- filteringTimeOut;
  This parameter comprises the following alternatives:

  - duration; or

  - stopTime.

- filteringCriteria;
  This parameter comprises the following alternatives:

  - serviceKey; or

  - addressAndService;

  - startTime;

  - extensions.

11.1.1.1.2   Result Parameters

None.

11.1.2   Invoking entity (SCF)

11.1.2.1   Normal procedure

SCF Precondition:

   (1) SLPI detects that service filtering has to be initiated at the SSF.

SCF Postconditions:

   (1) SLPI starts an application timer to monitor the expected end of service filtering.

   (2) The **SCME** is in the state "Waiting For ServiceFilteringResponse".

Sending the "ActivateServiceFiltering" operation causes a transition of the **SCME** from the state "Service Filtering Idle" to the state "Waiting For SSF Service Filtering Response". The **SCME** remains in this state until the application timer in the SLPI expires. The **SCME** is informed by the SLPI about timer expiration. Then it moves to the state "Service Filtering Idle".

If no errors occurred after receiving an "ActivateServiceFiltering" at the SSF an empty Return Result is sent to the SCF. That causes no state transition in the **SCME**.

To change the parameters of an existing service filtering entity the SCF has to send an "ActivateServiceFiltering" operation with the same "filtering Criteria". The second parameter set replaces the first one.

Sending sFTariffMessage in the "ActivateServiceFiltering" operation can be used by the service logic for charging of the service user. Using the sFTariffMessage for charging of the service user the SCF determines the charges, and it will act as charge determination point as described in ES 201 296.

The filtering characteristics parameter comprises two alternatives, *interval* or *numberOfCalls*.

In case the alternative *interval* is selected then the following procedure applies.

After expiration of the interval timer the next call to arrive causes following actions:

   • sending of an "InitialDP" operation;

   • sending of an "ServiceFilteringResponse";

   • starting again the interval timer.

When filtering is started the first interval is started.

In case the alternative *numberOfCalls* is selected the following procedure applies:

   • the number of calls matching the service filtering criteria is counted;

   • the n'th call causes an "InitialDP" operation and an "ServiceFilteringResponse" operation sent to the SCF. This threshold value is met if the sum of all counters assigned to one service filtering entity is equal to "numberOfCalls";

   • a number of calls of 0 indicate that none of the calls matching the filtering criteria will result in sending of an "InitialDP" operation and a "ServiceFilteringResponse" operation.

If ActivateServiceFiltering implies several counters, e.g. filtering on several dialled numbers, the numberOfCalls would include calls to all the dialled numbers.

When the *filteringTimeout* time expires, a "ServiceFilteringResponse" is sent to the SCF and service filtering is stopped.
The *filteringTimeOut* parameter comprises two alternatives, duration (relative time) or stopTime (date and time).

If "stopTime" was already met, i.e. the value of the stopTime is less than the value of the actual time but the difference does not exceed the value equivalent to 50 years, then service filtering is immediately stopped and the actual counter values are reported to the SCF. This occurs in cases where the SCF wishes to explicitly stop a running service filtering.

11.1.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors, are described in clause 15.

11.1.3   Responding entity (SSF)

11.1.3.1   Normal procedure

SSF Precondition:

   None.

SSF Postconditions:

   (1) The SSME-FSM is in the state "Non Call Associated Treatment".

   (2) A Return Result is sent after the successful "ActivateServiceFiltering" is executed.

If there is no already existing **SSME-FSM** for the "filteringCriteria" provided then a new **SSME-FSM** is created. This **SSME-FSM** enters the state "Non-Call Associated Treatment" and initializes the service filtering for the specified IN calls. The parameters "filteredCallTreatment", "filteringCharacteristics", "filteringCriteria", "filteringTimeOut" and "startTime" are set as provided in the operation. A number of counters will be allocated and reset. In the case of the "startTime" that has not been met yet, the service filtering will be started at the specified point in time. If "startTime" is not provided or was already met, the SSF starts filtering immediately.

If the operation "ActivateServiceFiltering" addresses an already existing service filtering entity the parameters "filteredCallTreatment", "filteringCharacteristics" "filteringTimeOut" and "startTime" are modified as provided in the operation. In the case that the addressed service filtering entity is active the SSF reports the counter values to the SCF via the operation "ServiceFilteringResponse". The service filtering process is stopped if an already expired "stopTime" or "duration" equal to ZERO or a new not yet met "startTime" is provided. The SSF then proceeds as described for "ServiceFilteringResponse". In the case of the "startTime" that has not been met yet, the service filtering will be continued at the specified point in time.

If the service filtering proceeds then the **SSME-FSM** remains in the state "Non-Call Associated Treatment". Otherwise the SSME-FSM moves to state "Idle Management".

When a call matches several active "filteringCriteria" it should be subject to filtering on the most specific criteria, i.e. the criteria with the longest "callingAddressValue" or "locationNumber", or alternatively the criteria with the largest number of parameters specified.

When performing service filtering with the "filteringCriteria" - "addressAndService" the first parameters checked will always be the "serviceKey" and "calledAddressValue".

If an "ActivateServiceFiltering" operation is passed to the SSF with the "filteringCriteria" "addressAndService" with both callingAddressValue and "locationNumber" present, the following is applicable.

When the SSF receives a call that matches "serviceKey" and "calledAddressValue" (in the active "filteringCriteria"), it investigates whether or not the locationNumber" is present in the initial address message. If it is present and matches the active "filteringCriteria" the call is filtered. If the SSF finds that the "locationNumber" is absent, then it will check the "callingAddressValue" and perform filtering depending on that parameter.

If no errors occurred after receiving an "ActivateServiceFiltering" on the SSF an empty Return Result is sent to the SCF. That causes no state transition in the **SSME-FSM**.

Following application timers are used:

- Detect moment to start service filtering (start time);

- Duration time for service filtering;

- Interval time for service filtering (for timer controlled approach).

On receipt of this operation with a request to send a chargingTariffInformation for subscriber charge in a ChargingMessage this is always mapped onto the ISUP-APM (Application Transport Message). The Application Context Identifier shall be set to "charging ASE". The ASF-parameter sFTariffMessage is mapped to the Encapsulated Application Information subfield.

- Handling of Application Transport Instruction Indicators (ATII).

  This cannot be transmitted over INAP, so the SSP shall set-up the values to be transmitted in ISUP. The provision of values 'release call' or 'continue call' is possible and which value is used is a network provider option.

- Handling of Acknowledgement Timer (Tcrga).

  In case, the parameter sFTariffMessage is included for sending a tariff the SSF has to take over the following signalling specific tasks of the charge determination point: Start the timer Tcrga awaiting the confirmation, respectively, representing the acknowledgement. On receipt of the acknowledgement the timer Tcrga shall be cancelled.

- The exceptional procedures:

  - on expiry of timer Tcrga; or

  - on receipt of a CRGT or AOCRG confirmation primitive with the indication "not accepted"; or

  - on receipt of the Charging_Error primitive;

  - have to be performed at the SSF (either releases the call or continues the call, this is a network provider option).

11.1.3.2 Error handling

If the SSF detects an error with any of the defined error values then this error is reported to the SCF.

The event is recorded in the SSF and an error condition indicated.

In case a new **SSME FSM** should be created, the relationship is ended and all concerned resources (e.g. counters) are released. The **SSME FSM** remains in the state "Idle Management".

In case there is already an existing **SSME FSM**, the service filtering data remains unchanged. The **SSME FSM** remains in the state "Non-Call Associated Treatment".

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors, are described in clause 15.

11.2 ActivityTest procedure

11.2.1 General description

This operation is used to check for the continued existence of a relationship between the SCF and SSF. If the relationship is still in existence, then the receiving entity will respond. If no reply is received within a given time period, then the SCF which sent this operation will assume that the receiving entity has failed in some way and will take the appropriate action.
As an option, this operation may be used in the reverse direction by the SSF to check for the continued existence of a relationship with the SCF.

11.2.1.1   Parameters

11.2.1.1.1   Argument Parameters

None.

11.2.1.1.2   Result Parameters

None.

11.2.2   Invoking entity (SCF)

11.2.2.1   Normal procedure

SCF Preconditions:

  (1) A relationship exists between the SCF and the SSF.

  (2) The activity test timer (Tati) expires, after which the "ActivityTest" operation is sent to the remote entity.

  (3) The **SCME** is in state "Activity Test Idle".

SCF Postcondition:

  (1) The **SCME** is in the state "Waiting for Activity Test Response". If a Return Result "ActivityTest" is received, the **SCME** resets the activity test timer, returns to state "Activity Test Idle", and takes no further action.

11.2.2.2   Error handling

If a time out on the "ActivityTest" operation or a P-Abort is received from TCAP, this is an indication that the relationship with the remote entity was somehow lost. If a time-out is received, SCF aborts the dialogue.

The SLPI that was the user of this dialogue will be informed, the corresponding **SCSM-FSM** will move to the state "idle".

11.2.3   Responding entity (SSF)

11.2.3.1   Normal procedure

SSF Precondition:

  (1) A relationship exists between the SCF and the SSF.

SSF Postconditions:

  (1) If the Dialogue ID is active and if there is a **FSM for CSA** or an **Assisting/Hand-off FSM** using the dialogue, the **SSME-Control** sends a Return Result "ActivityTest" to the SCF; or

  (2) If the Dialogue ID is not active, the TCAP in the SSF will issue a P-Abort; the **SSME-Control** will in that case never receive the "ActivityTest" req.ind and thus will not be able to reply.

11.2.3.2   Error handling

Operation related error handling is not applicable, due to class 3 operation.

11.2.4   Invoking entity (SSF)

11.2.4.1   Normal procedure

SSF Preconditions:

   (1) A relationship exists between the SCF and the SSF.

   (2) The activity test timer (Tati) expires, after which the "ActivityTest" operation is sent to the remote entity.

   (3) The **FSM for CSA** is in state "Active".

SSF Postcondition:

   (1) The **FSM for CSA** remains in the same state if a Return Result "ActivityTest" is received; the **FSM for CSA** resets the activity test timer, and takes no further action.

11.2.4.2   Error handling

If a time out on the "ActivityTest" operation or a P-Abort is received from TCAP, this is an indication that the relationship with the remote entity was somehow lost. If a time-out is received, the SSF aborts the dialogue. Whether or not the call survives beyond this error is based on triggering information.

The **CSA** that was using this dialogue is informed.

11.2.5   Responding entity (SCF)

11.2.5.1   Normal procedure

SCF Precondition:

   (1) A relationship exists between the SCF and the SSF.

SCF Postcondition:

   (1) The **SSME-Control** sends a Return Result "ActivityTest" to the SSF.

11.2.5.2   Error handling

Operation related error handling is not applicable, due to class 3 operation.

11.3   ApplyCharging procedure

11.3.1   General description

This operation is used for interacting from the SCF with the SSF charging mechanisms. The "ApplyChargingReport" operation provides the feedback from the SSF to the SCF.

This operation is used to request the SSF to perform call supervision based on a given credit limit. This "credit" limit can be expressed in cost, time and pulses. The call supervision can be related to a party (addressed via legID) or related to an SRF connection (addressed via callSegmentID).

A possibility exists for the ApplyCharging (AC) operation to be invoked on multiple occasions. AC can be applied during call setup of a leg or at User Interaction in order to request to start of call supervision. In addition AC can be applied when the leg or SRF is connected. In this case the AC is used to update the call supervision information.

The charging scenarios supported by this operation are B1 and B2 (refer to clause 16 "Charging Scenarios supported by Core INAP") in case the SSF is the Charge GEN point.

In case the basic network charging applies (scenario A, refer to clause 16 "Charging Scenarios supported by Core INAP"), the AC operation can be used for call supervision for cost and pulses if the basic network charging GEN is in the SSP.

11.3.1.1   Parameters

11.3.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- aChBillingChargingCharacteristics;

- sendCalculationToSCPIndication;

- aChChargingAddress.

    It comprises the following alternatives:

    - legID; or

    - srfCallSegment; or

    - bNCF.

- extensions;

- callSupervision.

This parameter comprises the following subparameters:

- supervisionMethod;
    this field comprises a choice of maximum cost, pulse or duration to supervise.

- warningBeforeLimitReached;

- releaseWhenLimitReached;

- reportConditions.

11.3.2   Invoking entity (SCF)

11.3.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The SLPI has determined that an "ApplyCharging" operation has to be sent.

(3) The **FSM for CS** is in any state except 'CS Control Idle'.

SCF Postconditions:

(1) No **FSM** state transition.

(2) The SLPI is expecting "ApplyChargingReport" operations from the SSF.

This operation is invoked by the SCF if a SLPI results in the request of interacting with the charging mechanisms within the SSF to get back information about the charging.

If the call supervision is activated, then a credit limit is granted by the AC operation to the charging process of the indicated aChChargingAddress. Following possibilities are applicable for the ApplyCharging call supervision:

1) The SCF determines the Tariff/AddOncharge.

   a) The FCI operation is used to determine a currency tariff. In this case the AC can indicate a call supervision limit based on currency.

   b) The FCI operation is used to determine a pulse tariff. In this case the AC can indicate a call supervision limit based on pulse units.

2) The SSF determines the Tariff/AddOncharge, instructed by the SCF or from a succeeding exchange. The same applies as in point 1.

3) The SCF determines the charging and evaluate a call duration limit: In this case the AC call supervision limit is based on duration.

4) The Basic Network Charging Function (BNCF), possibly controlled by the SCF using the SCI operation, determines the Tariff/AddOnCharging.

The Call Supervision is independent of charging recording (charging recording can be performed in SSF and/or SCF). The use of the FCI operation in combination with the AC operation is used to give charging related information to be able to supervise and not necessarily for charging recording in the SSF.

The SCF, in case the call is not released yet by the SSF (release when limit reached handling) or by signalling events, can instruct updates of the limit granted through new information in the AC.

11.3.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

The AC call supervision limit for BNCF based on pulse units or currency can only be used in case the BNCF in the SSP is based on the same meaning of the pulse units or the same currency. In case the currency used in the AC has not the same value as the currency used in the SSP by the supervised charging an error processing shall be performed. For pulses no error treatment can be performed in the SSP.

11.3.3   Responding entity (SSF)

11.3.3.1   Normal procedure

SSF Preconditions:

   (1) A relationship exists between the SCF and the SSF.

   (2) The **FSM for CS** in initiating SSF is in one of the following states:

   "Waiting for Instructions";
   "Waiting for End of User Interaction(WFI)";
   "Waiting for End of User Interaction(MON)";
   "Waiting for End of Temporary Connection(WFI)";
   "Waiting for End of Temporary Connection(MON)";
   "Monitoring"; or
   the **Assisting/Hand-off FSM** in assisting SSF is in state;
   "Waiting for Instructions".

SSF Postcondition:

   (1) No FSM for CS or Assisting/Hand-off FSM state transition.

The measurement for the credit limit is defined with the charging process applicable in the SSP (for example the FCI can specify a call attempt and/or a communication charge).

When the call supervision limit is reached in the SSF, a final ACR is transmitted to the SCF. Two conditions can occur: A final ACR is transmitted when the call (legID or callSegmentID as specified in the aChChargingAddress parameter) is still active (no release when limit is reached is given in the AC) or when the call is released (a release when limit is reached was given with the AC or the call was released by the user).

The ApplyChargingReport shall always report the exact used credit. Even in case of a final call active or a final call released report. Any add on charging that occurred at reaching almost the credit limit could exceed the current limit granted. When reporting ACR, the exact credit used shall be reported and not the allowed credit used.

In case of intermediate reporting requested and/or in case no release instruction is requested by the ApplyCharging operation, the SSF shall continue to measure the used credit for further call supervision possibilities instructed by the SCF, or for final reporting in case the call is released.

In case the releaseWhenLimitReached is specified by the SCF, the following rules apply to the release of the "connection":

- In case the parameter aChChargingAddress specifies a legID, this leg shall be released. In case there is only one remaining leg in the call segment of the released leg, all legs and resources are released. In case there is more then one remaining leg, the remaining legs are not released.

- In case the parameter aChChargingAddress specifies a callSegmentID, all legs shall be released within this call segment.

- The possible given warningDirection parameter does not influence the legID to be released.

The SSF will handle the release in accordance with the instructions received in the last AC.

11.3.3.2   Error handling

For CAMEL: TaskRefused: In addition to the generic error handling noted below, this error shall be indicated when:

- a previously received call period duration is pending,
- a tariffSwitchInterval is indicated when a previously received tariffSwitchInterval is pending.

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.4   ApplyChargingReport procedure

11.4.1   General description

This operation is used by the SSF to report charging related information to the SCF as requested by the SCF using the "ApplyCharging" operation.

During a connection configuration the "ApplyChargingReport" operation may be invoked on multiple occasions. For each call party and each connection configuration, the "ApplyChargingReport" operation may be used several times. Note that at least one "ApplyChargingReport" operation is to be sent at the end of the connection configuration charging process.

11.4.1.1   Parameters

11.4.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- callResult:
    This parameter comprises the following alternatives:

- callResult; or

- callSupervisionResult:
    This subparameter comprises the following fields:

    - callResult;

    - reportConditionInformation;

    - aChChargingAddress;

    - supervisionResult.
        This field comprises the following alternatives:

        - cost:
            This subfield comprises the following elements:

            - usedCurrencyFactor;

            - usedScale;

            - reportedCurrency; or

            - pulses:
                This subfield comprises the following elements:

                - usedPulseUnits; or

            - time:
                This subfield comprises the following elements:

                - usedDuration;

                - usedDurationAfterSwitchOverTime.

    - minLimitNeeded;

    - extensions.

11.4.2   Invoking entity (SSF)

11.4.2.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) A charging event has been detected that was requested by the SCF via an "ApplyCharging" operation.

SSF Postconditions:

(1) If the call segment of the aChChargingAddress is released the FSM moves to the "Idle" state if there is no other
    EDP armed and no report requests are pending. Otherwise, the FSM shall remain in the same state.

(2) If the call segment of the aChChargingAddress is not released the FSM shall remain in the same state.

This operation is invoked if a charging event has been detected that was requested by the SCF. The "ApplyChargingReport" operation only deals with charging events within the SSF itself.

The events related to call supervision are:

- intermediate report is invoked when the interval for reporting is reached.

- a final report is invoked when:

  - the leg or the SRF connection or the complete CS is released;

  - the credit limit is reached for the legID or callSegmentID;

- a final report is not invoked when the charging process applies for the specified aChChargingAddress is stopped by an ISP StopCharging message (see ES 201 296). The SCP has to use ENC/RNC to handle these messages for influencing the AC/ACR.

Reporting Rules for Pulse and Cost based Supervision.

1) Reporting of *UsedPulses/UsedCost* does not depend on the *ReportConditionInformation* parameter or the applied tariff model.

2) If at Answer the remaining *maximumPulseUnits/maximumTariffCurrency* are less then the pulses/currency needed then the sum of the CallSetup charges and the charges for the first tariff interval will be reported in the ACR *UsedPulses/UsedCost*. The reported *UsedPulses/UsedCost* can be greater than *maximumPulseUnits/maximumTariffCurrency*.

NOTE 1: Charges to be applied with answer(start of charge) will always be reported completely to avoid any misuse.

3) If after answer the limit is reached then ACR is send at the end of the last tariff interval (refer to ES 201 296) for that the limit was still large enough. *UsedPulses/UsedCost* has to contain all charges applied until this moment. It will not contain charges to be applied with the next following tariff interval.(If the call/leg is not released these charges will be reported with a subsequent ACR if the SCF provides new *maximumPulseUnits/maximumTariffCurrency*)

4) If the limit is not efficient for an addOnCharge(X = a lot) message received an ACR(with minLimitNeeded = X) is sent. In this ACR all charges already applied without the received addOnCharges will be reported within parameter supervisionResult. The ackknowledge of the addOnCharge is delayed until the response from the SCF. If the call/leg is not released the ackknowledge is send. The addOnCharges will be reported with a subsequent ACR if the SCF provides new *maximumPulseUnits maximumTariffCurrency*.

NOTE 2: X is the value of the parameter addOnCharge.

5) In case of charge registration in the SSP the data in the corresponding Call Data Records corresponds to the reported *UsedPulses/UsedCost* (cumulated).

EXAMPLE 1: Report Used currency:
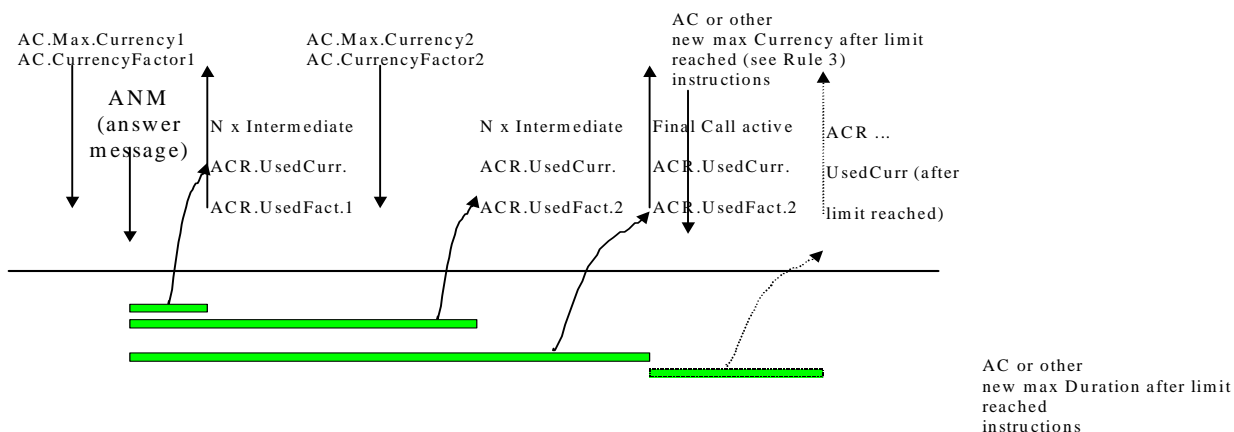
EXAMPLE 2:   Report Used Duration:
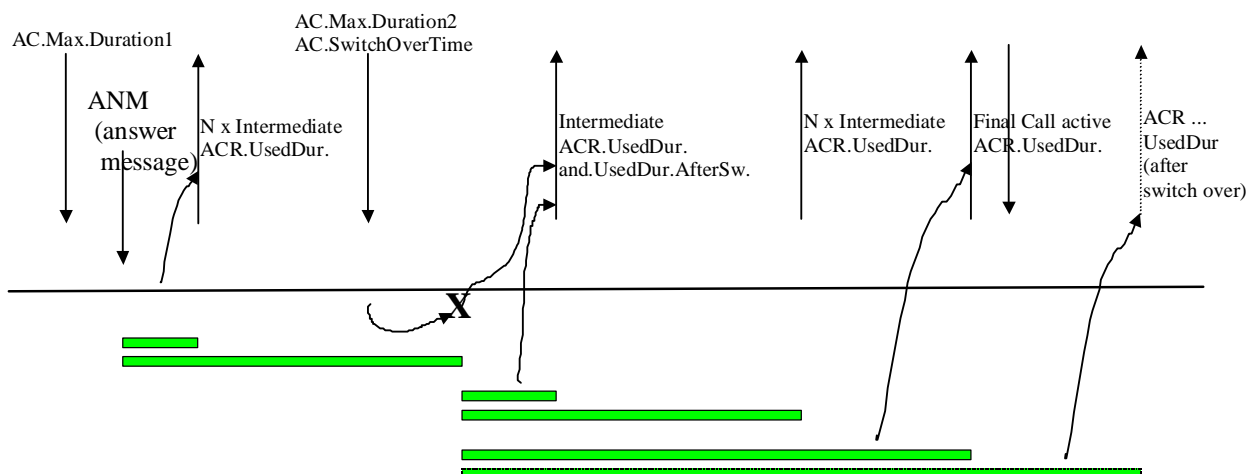
11.4.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.4.3   Responding entity (SCF)

11.4.3.1   Normal procedure

SCF Preconditions:

   (1) A relationship exists between the SCF and the SSF.

   (2) An "ApplyCharging" operation has been sent at the request of an SLPI and the SLPI is expecting an "ApplyChargingReport" from the SSF.

SCF Postcondition:

   (1) No **FSM for CS** state transition if further reports, including "EventReportBCSM" and "CallInformationReport", are expected, or Transition to the state "Idle" if the report is the last one and no "EventReportBCSM" or "CallInformationReport" is expected.

On receipt of this operation the SLPI, which is expecting this operation will continue processing.

11.4.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 the TCAP services used for reporting operation errors are described in clause 15.

11.5   AssistRequestInstructions procedure

11.5.1   General description

This operation is sent to the SCF by an SSF, which is acting as the assisting SSF in an assist or hand-off procedure, or by a SRF. The operation is sent when the assisting SSF or SRF receives an indication from an initiating SSF containing information indicating an assist or hand-off procedure.

Refer to EN 301 931-3 for further details on the assist and hand-off procedures.

11.5.1.1   Parameters

11.5.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- correlationID;

- iPAvailable;

- iPSSPCapabilities;

- extensions.

11.5.2   Invoking entity (SSF)

11.5.2.1   Normal procedure

SSF Precondition:

(1)   An assist indication is detected indicating an Assist or Hand-off procedure in assisting SSF.

SSF Postcondition:

(1)   The assisting SSF waits for instructions.
The **Assisting/Hand-off FSM** is in the state "Waiting For Instructions".

On receipt of an assist indication from the initiating SSF, the SSF or SRF shall assure that the required resources are available to invoke an "AssistRequestInstructions" operation in the SSF/SRF and indicate to the initiating SSF that the call is accepted (refer to ITU-T Recommendation Q.71). The "AssistRequestInstructions" operation is invoked by the SSF or SRF after the call, which initiated the assist indication, is accepted. The **FSM for Assisting SSF**/**FSM for Handed-off SSF** transitions to state "Waiting For Instructions".

11.5.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.5.3   Responding entity (SCF)

11.5.3.1   Normal procedure

SCF Preconditions:

(1)   A relationship exists between the SCF and the initiating SSF in case of assist procedure.

(2)   The SCF waits for "AssistRequestInstructions".

SCF Postcondition:

(1)   An SSF or SRF instruction is being prepared.

On receipt of this operation in the **SCSM** state "Waiting for Assist Request Instructions", the SCP has to perform the following actions:

If the "AssistRequestInstructions" operation was received from an assisting SSF, and the resource is available, the SCSM prepares the "ConnectToResource" and the SRF User Interaction operations (e.g. "PlayAnnouncement" or "PromptAndCollectUserInformation", see ITU-T Recommendation Q.1238.3) to be sent to the assisting SSF.

The SCF determines SSF/SRF by means of "correlationID", "destinationNumber" or network knowledge.

If the "AssistRequestInstructions" operation was received from a SRF, and the resource is available, the SCSM prepares the User Interaction operation to be sent to the SRF.

On receipt of this operation from the Hand-off SSF, the **SCSM** associated with the Hand-off SSF transits from the "Idle" state to the "Preparing SSF Instructions" state.

11.5.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.6   CallGap procedure

11.6.1   General description

This operation is used to request the SSF to reduce the rate at which specific service requests are sent to the SCF. The operation applies to a request for filtering of service request (i.e. as opposed to filtering of calls).

The CallGap operation can be received inside as well as outside a call context transaction.

11.6.1.1   Parameters

11.6.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

> gapCriteria;
> It consists of the following alternatives:
>
> - basicGapCriteria; or
>
> - compoundGapCriteria;
>
> ```
> The compound criteria consists of the scfID and the basicGapCriteria.
> ```
>
> gapIndicators;
> This parameter comprises the following subparameters:
>
> - duration;
>
> - gapInterval;
>
> - controlType;
>
> - gapTreatment;
>   This parameter comprises the following alternatives:
>
>   - informationToSend; or
>
>   - releaseCause; or
>
>   - both;
>
> - extensions.

11.6.2   Invoking entity (SCF)

11.6.2.1   Normal procedure

SCF Preconditions:

(1) The SCF detects an overload condition persists and IN service request gapping has to be initiated at the SSF; or

(2) the SCF receives a manually initiated service gapping request.

SCF Postcondition:

(1) The **SCME FSM** remains in the same state upon issuing the "CallGap" operation.

A congestion detection and control algorithm monitors the load of SCP resources. After detection of a congestion situation the parameters for the "CallGap" operation are provided.

If the congestion level changes new "CallGap" operations may be sent for active gap criteria but with new gap interval. If no congestion is detected gapping may be removed.

A manual initiated call gap will take prevail over an automatic initiated call gap.

11.6.2.2   Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.6.3   Responding entity (SSF)

11.6.3.1   Normal procedure

SSF Preconditions:

(1) Service request gapping for gapCriteria is not active; or

(2) Service request gapping for gapCriteria is active.

SSF Postconditions:

(1) The **SSME-FSM** is in the state "Non call associated treatment".

(2) Service request gapping for gapCriteria is activated; or

    Service request gapping for gapCriteria is renewed; or

    Call gapping for gapCriteria is removed.

If there is no already existing **SSME-FSM** for the gap criteria provided, a new **SSME-FSM** is created.

If no compound gapcriteria (basic gap criteria and SCFID) is provided, this refers in general to the SSME-FSM without a SCFAddress. **This SSME-FSM** enters the state "Non call associated treatment" and initializes service request gapping for the specified IN calls. The parameters "gapIndicators", "controlType", "gapTreatment" and scfID for the indicated gap criteria will be set as provided by the "CallGap" operation.

If gapTreatment parameter is not present, the SSF will use a default treatment depending on network operator implementation. The default treatment can e.g. be either to release call or continue call processing.

In case both manually initiated and automatically initiated service request gapping are active for the same "gapCriteria", the manuallyInitiated call gapping will prevail over automatically initiated ("sCPOverloaded"). More specifically, the following rules will be applied in the SSF to manage the priority of different control Types associated with the same "gapCriteria":

- If an **SSME-FSM** already exists for the "gapCriteria" provided, then:

1) If the (new) "controlType" equals an existing "controlType", then the new parameters (i.e. "gapIndicators" and "gapTreatment") will overwrite the existing parameter values.

2) If the (new) "controlType" is different than the existing "controlType", then the new parameters (i.e. "controlType", "gapIndicators", and "gapTreatment") will be appended to the appropriate **SSME-FSM** (in addition to the existing parameters). The **SSME-FSM** remains in the state "Non-Call Associated Treatment".

If the SSF meets a TDP, it checks if service request gapping was initiated for the same SCF (compound gap criteria, i.e. scfID and basicGapCriteria) as the one currently assigned to this TDP or if call gapping exists with no provided scfID (basicGapCriteria, no SCFID).

The SSF will for basicGapCriteria check if service request gapping was initiated either for the "serviceKey" or for the "calledAddressValue" assigned to this TDP. If not, an "InitialDP" operation can be sent. In case service request gapping was initiated for "calledAddressAndService"; or "callingAddressAndService" and the "serviceKey" matches, a check on the "calledAddressValue" and "callingAddressValue" - and optionally "locationNumber" - for active service request gapping is performed. If not, an "InitialDP" operation can be sent.

In case of gapping on "callingAddressAndService" and the parameter "locationNumber" is present, gapping will be performed on "locationNumber" instead of "callingAddressValue".

If a call to a controlled number matches only one "gapCriteria", then the corresponding control is applied.
If both "manuallyInitiated" and "sCPOverload" controls are active, then only the manually initiated control will be applied.

If a call matches several active "basicGapCriteria", then the treatment as specified in the CallGap associated with the gapCriteria with the highest priority should be applied, with the priority being from high to low:

1) calledAddressAndService/calledAddressValue;

2) callingAddressAndService;

3) gapOnService;

4) gapAllINTraffic.

For example, a call with called number 123456 and ServiceKey = NP matches two CallGaps, one with gapCriteria 'CalledAdressValue=123' and another with 'gapOnService=NP'. Then the call is subject to the control of the service request CallGap with 'CalledAdressValue=123'.

In case multiple call gapping procedures are active with the same gap criteria, the "manuallyInitiated" service request gapping shall prevail over automatically initiated service request gapping ("sCPOverloaded").

If a call to a controlled called number or from a controlled calling number matches several active "gapCriteria" of the same type (in this context 'calledAddressAndServic' and 'calledAddressValue' are seen as one type), then only the "gapCriteria" associated with the longest called respective calling party number should be used, and the corresponding control should be applied.
For example, the codes 1234 and 12345 are under control. Then the call with 123456 is subject to the control on 12345.

If a call to a controlled called number matches calledAddressAndService and calledAddressValue with the same number length, than calledAddressAndService has priority. Furthermore, if both "manuallyInitiated" and "sCPOverloaded" "controlTypes" are active for this "gapCriteria", then the "manuallyInitiated" control will be applied.

If service request gapping is performed on a call for a particular service and triggering of this service is allowed, apart from the case that "gapAllINTraffic" is active, no other gap criteria should be applied to the same service.

If "gapAllINTraffic" is active, then the checks for other criteria will be applied as described above. After these checks, control according to "gapAllINTraffic" will be applied for every IN call not blocked by other active criteria.

Active GapCriteria with assigned scfID (i.e. compoundGapCriteria) will have higher priority than the others. In case an entry with SCFID matching the current call exist all other criteria without scfID are not evaluated.

The matching entries with scfID are evaluated in accordance with the priority rules for the basic criteria listed above.

If service request gapping shall be applied and there is no service request gap interval active, an "InitialDP" can be sent including the "cGEncountered" parameter according to the specified controlType. A new possible service request interval will be initiated as indicated by "gapInterval".

If a service request interval is active, no "InitialDP" operation is sent and the call is treated as indicated by "gapTreatment".

The service request gap process is stopped if the indicated duration equals ZERO.

If the call gapping proceeds then the **SSME-FSM** remains in the state "Non call associated treatment". Otherwise, the **SSME-FSM** moves to state "idle management".

11.6.3.2   Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.7   CallInformationReport procedure

11.7.1   General description

This operation is used to send collected specific call information for a single call/call party (identified by leg id) to the SCF as requested by the SCF in previous "CallInformationRequest" operation. The report is sent at the end of a call/call party connection which is indicated by one of the events specified below. This operation is not backwards compatible with ITU-T Recommendation Q.1214 in the case where the A-party is released and the B-party is kept (e.g. A-Party DP-Disconnect armed in interrupt mode).

11.7.1.1   Parameters

11.7.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- requestedInformationList;

- extensions;

- legID;

When absent, it shall apply to the default leg, i.e. the default legID of the "outgoing" leg (i.e. the passive leg in an O-BCSM or the controlling leg in an T-BCSM). The default leg within the initial Call Segment this can be a leg created by InitiateCallAttempt; or Connect/Continue/ContinueWithArgument.

11.7.2   Invoking entity (SSF)

11.7.2.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) "CallInformationReport" is pending due to a previously received "CallInformationRequest" operation.

(3) The indicated or default party is released from the call (e.g. due to user or SCF initiated release) or the call setup towards the indicated or default party is not completed (e.g. due to signalling events like busy, network congestion).

(4) Requested call information has been collected.

SSF Postconditions:

(1) The **FSM for CS** in the SSF shall move to the "Idle" state in the case where no other report requests are pending and no EDPs are armed otherwise the FSM for CS shall remain in the same state. See tables 19 and 20.

(2) When the CallInformationReport is sent due to the receipt of the operation DisconnectLeg, which was followed by a state change to Waiting for Instructions, the **FSM for CS** remains in the same state ("Waiting for Instructions") independent of other pending reports or armed EDPs.

If the FSM for CS executes a state transition caused by one of the following events:

- Release for the indicated or default leg;

- Abandon for the indicated or default leg;

- Busy for the indicated or default leg;

- SSF no answer timer expiration for the indicated or default party;

- Route select failure indicated by the network for the indicated or default leg;

- Release of the indicated or default leg by the SCF (DisconnectLeg);

- Release of call or call segment initiated by the SCF (ReleaseCall);

and "CallInformationRequest" is pending for the indicated or default leg, then one "CallInformationReport" operation is sent for each such leg to the SCF containing all information requested.

If a "CallInformationReport" has been sent to the SCF then no "CallInformationReport" is pending, i.e. a further "CallInformationReport", for example in the case of follow-on, has to be explicitly requested by the SCF.

If an event causing the "CallInformationReport" is also detected by an armed EDP-R then immediately after "CallInformationReport" the corresponding "EventReportBCSM" has to be sent. See tables 19 and 20.

If an event causing the "CallInformationReport" is also detected by an armed EDP-N then immediately before "CallInformationReport" the corresponding "EventReportBCSM" has to be sent. See tables 19 and 20.

11.7.2.2   Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.7.3   Responding entity (SCF)

11.7.3.1   Normal procedure

SCF Preconditions:

    (1) A relationship exists between the SCF and the SSF.

    (2) An SLPI is expecting "CallInformationReport".

SCF Postcondition:

    (1) The SLPI may be further executed.

In any state (except "Idle") the **SCSM** may receive "CallInformationReport" from the SSF, when the "CallInformationReport" is outstanding.

If "CallInformationReport" is outstanding and the service logic program indicates that the processing has been completed, the SCSM remains in the same state until it receives the "CallInformationReport" operation.

When the SCF receives the "CallInformationReport" operation and the service logic processing has been completed, then the **FSM for CS** in the **SCSM** moves to the "Idle" state. When the SCF receives the "CallInformationReport" operation and the service logic processing has not been completed yet, then the **SCSM** remains in the same state (EventReportBCSM and/or ApplyChargingReport and/or CallInformationReport pending).

11.7.3.2   Error handling

If requested information is not available, a "CallInformationReport" will be sent, indicating the requested information type, but with "RequestedInformationValue" filled in with an appropriate default value as specified by the network operator.

Operation related error handling is not applicable, due to class 4 operation.

11.8   CallInformationRequest procedure

11.8.1   General description

This operation is used to request the SSF to record specific information about a single call/call party (indicated by leg id) and report it to the SCF using the "CallInformationReport" operation.

11.8.1.1   Parameters

11.8.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

* requestedInformationTypeList;

* extensions;

* legID.

11.8.2   Invoking entity (SCF)

11.8.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The SLPI has determined that a "CallInformationRequest" operation has to be sent by the SCF.

SCF Postcondition:

(1) The SLPI is expecting a "CallInformationReport" from SSF.

When the service logic program requests call information, the SCF sends the "CallInformationRequest" operation to the SSF to request the SSF to provide call related information.

The "CallInformationRequest" operation specifies the information items to be provided by the SSF.

11.8.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.8.3   Responding entity (SSF)

11.8.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between SSF and SCF.

(2) The **FSM for CS** is in state "Waiting for Instructions".

SSF Postconditions:

(1) Requested call information is retained by the SSF.

(2) The SSF is waiting for further instructions, that is the **FSM for CS** remains in state "Waiting for Instructions".

The SSF may receive the "CallInformationRequest" operation within an existing call associated (CA) dialogue only.

The SSF allocates a record for the indicated or default party and stores the requested information if already available and prepares the recording of information items, that will become available later like for example "callStopTimeValue".

Call information may be requested for any call party connection (identified by a legID). The indicated leg may be any controlling leg or passive leg.

11.8.3.2   Error handling

In any other than the "Waiting for Instruction" state the "CallInformationRequest" operation will be handled as an error with the error code "UnexpectedComponentSequence".

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.9   Cancel procedure

11.9.1   General description

The SCF uses this class 2 operation to request the SSF to cancel outstanding requests, i.e. all active requests for EDP reports, "ApplyChargingReport" and "CallInformationReport" should be cancelled. The "Cancel" operation can be used to cancel all outstanding requests for either a specific call segment or all call segments (enables the termination of the SSF-SCF relationship as the SSF state machine goes to Idle).

> NOTE: The SCF may either invoke this operation to the SSF or to the SRF, different conditions will prevail in each case. For the usage of this operation for the SRF refer to EN 301 931-3.

11.9.1.1   Parameters

11.9.1.1.1   Argument Parameters

The operation argument consists of the following alternatives. These parameters are defined in clause 12.

- invokeID;
  (only applicable for SRF, refer to EN 301 931-3); or

- allRequests; or

- allRequestsForCallSegment; or

- callSegmentToCancel;
  (only applicable for SRF, refer to EN 301 931-3).

11.9.2   Invoking entity (SCF)

11.9.2.1   Normal procedure

SCF Preconditions:

  (1) A relationship exists between the SCF and the SSF.

  (2) An SLPI has determined that it is no longer interested in any reports or notifications from the SSF and that the relationship should be ended.

SCF Postcondition:

  (1) The SLPI remains in the "Waiting for Response from SRF" state; or
      in case all request for all call segments are cancelled, the relationship with the concerned FE (SSF) is ended and the **SCSM FSM** returns to "Idle" state.

11.9.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.9.3   Responding entity (SSF)

In case of Cancel(allRequests) or Cancel(allRequestsForCallSegment) the SSF is the responding entity.

11.9.3.1   Normal procedure

SSF Preconditions:

> (1) A relationship exists between the SCF and the SSF.

> (2) The **FSM for CS** is in the state "Waiting for Instructions" or "Monitoring".

SSF Postconditions:

> (1) All active requests for reports and notifications have been cancelled for either all call segments or a specific call segment.

> (2) In case the **FSM for CS** was in state "Monitoring" it shall return to idle; or
> In case the **FSM for CS** was in state "Waiting for Instructions" it will remain in that state. A subsequent call-processing operation (e.g. Connect, ContinueWithArgument) will move the **FSM for CS** state to "Idle". The call, if in active state, is further treated by SSF autonomously as a normal (non-IN-) call.

11.9.3.2   Error handling

Sending of return error on cancel is applicable in the cancel "allRequests" and "allRequestsForCallSegment" case. Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.10   CancelStatusReportRequest Procedure

11.10.1 General description

This operation is used to request the SSF to cancel a previous request to monitor the busy/idle status of a physical termination resource.

11.10.1.1   Parameters

11.10.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- resourceID;
  This parameter comprises the following alternatives:

  - lineID; or

  - facilityGroupID; or

  - facilityGroupMemberID; or

- trunkGroupID;
  extensions.

11.10.2 Invoking entity (SCF)

11.10.2.1   Normal procedure

SCF Precondition:

(1) The **SCME** is in state "Waiting for SSF Resource Status Report".

SCF Postcondition:

(1) The **SCME** moves to the same state.

11.10.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.10.3 Responding entity (SSF)

11.10.3.1   Normal procedure

SSF Precondition:

(1) The **SSME** is in the state "Non-Call Associated Treatment".

SSF Postcondition:

(1) **SSME** sends a Statusreport operation with the reportCondition parameter set to "cancelled" and moves to the state "Idle Management".

11.10.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.11   CollectInformation procedure

11.11.1 General description

The CollectInformation is a class 2 operation which is used by the SCF to request the call to return to the Collect_Information PIC, and then perform the basic originating call processing actions associated with this PIC (e.g. the checking of information in the CalledPartyNumber parameter with the supported dialling plan). This operation uses only the resources of the CCF/SSF to collect the information. The use of this operation is only appropriate for a call which have not yet left the setup phase.

When the user provides calledPartyNumber, Collect_Information PIC processing includes collecting of destination information from a calling party.

11.11.1.1   Parameters

11.11.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- extensions.

11.11.2 Invoking entity (SCF)

11.11.2.1   Normal procedure

SCF Preconditions:

   (1) A relationship exists between the SSF and the SCF.

   (2) An SLPI has determined that more information from the calling party is required to enable processing to proceed.

SCF Postcondition:

   (1) SLPI execution is suspended pending receipt of dialled digits.

This operation is invoked in the **SCSM FSM** state "Preparing SSF Instructions" if the SLP requires additional information to progress the call. It causes a transition of the **FSM** to the state "Waiting for Notification or Report".

11.11.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.11.3 Responding entity (SSF)

11.11.3.1   Normal procedure

SSF Preconditions:

   (1) A relationship exists between the SSF and the SCF.

   (2) **FSM for CS** is in state "Waiting for Instructions".

   (3) An operation "RequestReportBCSMEvent" for arming DP Collected_Information has been received.

SSF Postconditions:

   (1) The SSF has executed a transition to the state "Monitoring".

   (2) The SSF performs the call processing actions to collect destination information from the calling party. This may include prompting the party with in-band or out-band signals.

   (3) Basic call processing is resumed at PIC Collect_Information.

The SSP has to perform the following actions:

   • The SSF cancels T_{SSF}.

   • When DP Collected_Information will be encountered, an "EventReportBCSM" operation will be invoked, and the **FSM for CS** will return to the state "Waiting for Instructions".

11.11.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.12   Connect procedure

11.12.1 General description

This operation is used to request the SSF to perform the call processing actions to route a call to a specific destination. To do so, the SSF may use destination information from the calling party (e.g. dialled digits) and existing call set-up information (e.g. route index to a list of trunk groups) depending on the information provided by the SCF.

In general all parameters which are provided in a Connect operation to the SSF shall replace the corresponding signalling parameter in the CCF, if the relevant parameter has already been received in the CCF, and shall be used for subsequent call processing. For the CAMEL T-BCSM the information will not be used subsequently. Parameters which are not provided by the Connect operation shall retain their value (if already assigned) in the CCF for subsequent call processing.

11.12.1.1   Parameters

11.12.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- destinationRoutingAddress;

- alertingPattern;

- correlationID;
  Only allowed if the correlationID and sCFID are not embedded in the "destinationRoutingAddress";

- cutAndPaste;

- iSDNAccessRelatedInformation;

- originalCalledPartyID;
  The use of this parameter in the context of the "Connect" operation is to be specified by the network operator. For CAMEL this parameter carries the dialled digits, if the call is forwarded by the SCF;

- routeList;

- scfID;
  Only allowed if the scfID and correlationID is not embedded in the "destinationRoutingAddress";

- extensions;

- carrier;

- serviceInteractionIndicators;

- callingPartyNumber;

- callingPartysCategory;

- redirectingPartyID;

- redirectionInformation;

- displayInformation;

- forwardCallIndicators;

- genericNumbers;

- serviceInteractionIndicatorsTwo;

- iNServiceCompatibilityResponse;

- forwardGVNS;

- backwardGVNS;

- chargeNumber;

- callSegmentID;
  When not provided, a default CSID of 1 is assumed;

- legToBeCreated;
  When not provided, a default LegID of 2 is assumed;

- sDSSinformation;

- calledDirectoryNumber;

- bearerCapability;

- oCSIApplicable;
  Support for CAMEL;

- suppressionOfAnnouncement;
  Support for CAMEL;

- cug-Interlock;
  Support for CAMEL;

- cug-OutgoingAccess;
  Support for CAMEL;

- na-OliInfo;
  Support for CAMEL.

11.12.2 Invoking entity (SCF)

11.12.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) An SLPI has determined that a "Connect" has to be sent by the SCF.

NOTE: When a Connect operation is sent the SCF should ensure that LegToBeCreated does not correspond to an existing joined passive leg.

SCF Postcondition:

(1) SLPI execution may continue.

In the **SCSM FSM** state "Preparing SSF Instructions", this operation is invoked by the SCF if the service logic results in the request to the SSF to route a call to a specific destination. If no event monitoring has been requested and no reports (CallInformationReport and ApplyChargingReport) have been requested in a previously sent operation, a SCSM FSM transition to state "Idle" occurs. Otherwise, if event monitoring has been requested or any report (CallInformationReport and ApplyChargingReport) has been requested, the **SCSM FSM** transitions to state "Waiting for Notification or Report". When the "Connect" operation is used in the context of a hand-off procedure, the **SCSM FSM** transitions to state "Idle". However, in this case, the SCF must maintain sufficient information in order to correlate the subsequent "AssistRequestInstructions" operation (from the assisting SSF or SRF) to the existing SLPI.

11.12.2.2   Error handling

If reject or error messages are received, then the **SCSM** informs the SLPI and remains in the state "Preparing SSF Instructions".

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.12.3 Responding entity (SSF)

11.12.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) BCSM: Basic call processing has been suspended at a DP.

(3) The **FSM for CS** is in the state "Waiting for Instructions".

(4) The **CSCV** is in a valid connection view state according to CSCV transition table 22.

SSF Postconditions:

(1) The SSF performs the call processing actions to route the call to the specified destination.

(2) In the O-BCSM, when destinationRoutingAddress and no RouteList parameters are present in the Connect operation, then call processing resumes at PIC Analyse_Information.

(3) In the O-BCSM, when destinationRoutingAddress and RouteList parameters are present in the Connect operation, then call processing resumes at PIC Select_Route.

(4) In the T-BCSM for CAMEL forwarding a chained O-BCSM is used if the subscriber has an active O-CSI or there is an active N-CSI or there is an active D-CSI and the triggering criteria are satisfied and the last Connect operation included the "O-CSI applicable" flag.

(5) If no EDPs have been armed and neither a CallInformationReport nor an ApplyChargingReport has been requested, the FSM goes to state "Idle" . Otherwise, the **FSM for CS** goes to state "Monitoring".

(6) When the Connect operation is received in the following **CSCV** states: Terminating_Setup, Stable_1 Party, then a new O_BCSM shall be created (at O-Null PIC) and chained to the existing O/T_BCSM.
The existing O/T_BCSM shall pass the information available (e.g. bearerCapability) to the new O_BCSM. The call processing shall be resumed as defined.

NOTE: When a Connect operation is received the T_BCSM may be suspended in one of the following DPs: Termination_Attempt, Terminating_Attempt_Authorized, Termination_Attempt_Denied, Facility_Selected_and_Available, T_No_Answer, T_Busy and T_Disconnect - and the T_BCSM moves to the Present_ Call PIC.
When the CallProgressInd (bptyAlerted) or the SetupResp (answer) or the ReleaseReq signals are received from the called destination (C-Party) then the T_BCSM moves to the corresponding DP.
No events will be reported to the SLPI from the T_BCSM for that case where the controlling leg becomes surrogate in the **CSCV** state. By resumption of the DP the corresponding PIC is performed.

(7) When the Connect operation is received in the '1_Party' **CSCV** state and the controlling leg is connected to an O_BCSM, then the suspended O_BCSM becomes associated with the leg to be created.
When the controlling leg is connected to a T_BCSM, an error procedure is invoked.

(8) When the Connect operation is received in the 'Forward' **CSCV** state then the O_BCSM is associated with the leg to be created.

Table 22a: Transition for CS

CSCV State (original state): ⇒ Operation: ⇓	Originating Setup	Originating 1-Party Setup	Stable 1-Party	Terminating Setup	1-Party
Connect	Stable 2-Party (notes 1 and 3)	Error (Originating 1-Party Setup) note 7 or Stable 1-Party (notes 1 and 2)	Forward (Stable_Multi_Passive_Party: note 1)	Forward (Stable_Multi_Passive_Party: note 1) or Error (Terminating Setup) note 5	Originating Setup (Stable 2-Party note 1) or Error (1-Party) note 6

Table 22b: Transition for CS

CSCV State (original state): ⇒ Operation: ⇓	Stable 2-Party	Forward	Stable_Multi_Passive_Party	Stable Multi-Party
Connect	Error (Stable 2-Party)	Forward note 2 (Stable_Multi_Passive_Party note 1)	Error (Stable_Multi_Passive_Party)	Error (Stable Multi-Party)

NOTE 1: State transition occurs not until DP O_Term_Seized (i.e. when the CallProgress.req.ind (bptyAlerted) is received). If no such DP is encountered the transition is detected at DP O_Answer (i.e. as Setup.resp.conf (answer) is received from the called destination (automatic answer case)).

NOTE 2: In case of an unsuccessful call setup the Connect operation is allowed when call processing is suspended at the following DPs for leg p: O_Called_Party_Busy, Authorize_Route_Failure, Route_Select_Failure, O_No_Answer, O_Disconnect.

NOTE 3: Not allowed in O_Mid_Call DP (i.e. after entering Send_Call PIC).

NOTE 4: Void.

NOTE 5: Only allowed if call processing is suspended at the following DPs: Facility_Selected_And_Available, Termination_Attempt, Termination_Attempt_Authorized, T_Busy, T_No_Answer.

NOTE 6: The Connect operation is only allowed in the 1_Party CSCV state when the controlling leg is connected to an O_BCSM, when the controlling leg is connected to a T_BCSM an error procedure is invoked.

NOTE 7: An error TaskRefused is provided if Connect (and not Continue/ContinueWithArgument) is received to resume call setup for a leg to be created following the InitiateCallAttempt.

On receipt of this operation in the **FSM for CS** state "Waiting for Instructions", the SSP performs the following actions:

- The SSF cancels T_{SSF}.

- If "cutAndPaste" is present, then the SSF deletes ("cut") from the dialled IN number the indicated number of digits and pastes the remaining dialled digits at the end of the "destinationRoutingAddress" parameter delivered by the SCF. The resulting directory number is used for routing to complete the related call.

- If "cutAndPaste" is not present, then the "destinationRoutingAddress" parameter delivered by the SCF is used for routing to complete the related call. Note that in the case of hand-off, this results in routing to an assisting SSP or IP.

- If the "callingPartyNumber" is supplied, this value may be used for all subsequent SSF processing.

No implicit activation or deactivation of DPs occurs.

Statistic counter(s) are not affected.

Connect completes when the INAP processing of the operation is complete and before the SSP starts the processing necessary to select a circuit.

Therefore in order to detect route select failure after a "Connect" it is necessary to explicitly arm the "Route Select Failure" EDP before sending the "Connect" (although they may be in the same message).

11.12.3.2   Error handling

NOTE 1: When the Connect operation is received in an inappropriate state, a TaskRefused error shall be returned.

NOTE 2: When a Connect operation is received with a bearerCapability parameter value that is in conflict with the applied value for the addressed call segment and the procedures defined in ISUP, an "UnexpectedDataValue" error shall be returned.

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.13   ConnectToResource procedure

11.13.1 General description

This operation is used to connect a call from the SSF to a specialized resource. After successful connection to the SRF, the interaction with the caller can take place. The SSF relays all operations for the SRF and all responses from the SRF.

The ConnectToResource may either be sent to an initiating SSF or an Assisting-Hand-off SSF.

11.13.1.1   Parameters

11.13.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- resourceAddress;
  It consists of the following alternatives:

  - iPRoutingAddress;
    It is only valid when used in a single call segment CSA or at the Assist/Handed-off SSF. The subsequent user interaction(s) shall apply to this call   segment, i.e. to all parties connected to it; or

  - legID;
    Ignored for Assist/Handed-off SSF, if received; or

  - iPAddressAndLegID;
    Ignored for Assist/Handed-off SSF, if received; or

  - none;
    It is only valid when used in a single call segment CSA or at the Assist/Handed-off SSF. The subsequent user interaction(s) shall apply to this call   segment, i.e. to all parties connected to it; or

  - callSegmentID;
    Ignored for Assist/Handed-off SSF, if received; or

  - iPAddressAndCallSegment;
    Ignored for Assist/Handed-off SSF, if received;

- extensions;

- serviceInteractionIndicators;

- serviceInteractionIndicatorsTwo;

- uSIServiceIndicator;

- uSIInformation.

11.13.2 Invoking entity (SCF)

11.13.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The SLPI has determined that additional information from the call party is needed.

(3) The **FSM for CS** is in the state "Routing to Resource", substate "Determine Mode".

(4) The SLPI has determined that the SRF can be accessed from the SSF.

SCF Postconditions:

(1) The **SCSM** sends out a valid UI operation, e.g. "PlayAnnouncement", "PromptAndCollectUserInformation" or PromptAndReceiveMessage operation accompanying the "ConnectToResource".

(2) The **SCSM-FSM** moves to the state "User Interaction" (substate 'Suspended' or 'Not Suspended').

NOTE: In substate 'Not Suspended' only PlayAnnouncement is possible.

11.13.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.13.3 Responding entity (SSF)

11.13.3.1   Normal procedure

SSF Preconditions:

(1) A relationship has been established.

(2) The **FSM for CS** (initiating SSF) is in the state "Waiting for Instructions" or "Monitoring"; or **Assisting/hand-off -FSM** is in the State "Waiting for Instructions".

NOTE 1: During monitoring state it is only possible to perform user interaction in order to send tones, announcements or display information.

SSF Postconditions:

(1) The call is switched to the SRF.

(2) A relationship to the SRF is established.

(3) If in state "Waiting for Instructions" the **FSM for CS** (initiating SSF) moves to the state "Waiting for End of User Interaction (WFI)". If necessary (for CAMEL mandatory)", T_{SSF} is set; or
If in state "Waiting for Instructions" the **Assisting/hand-off FSM** (assisting SSF) moves to the state "Waiting for End of User Interaction.(WFI)". If necessary (for CAMEL mandatory)" T_{SSF} is set.

(4) If in state "Monitoring" the **FSM for the CS** moves to the state "Waiting for End of User Interaction (MON)". If necessary (for CAMEL mandatory)" a guard timer T_{SSF} is set.

NOTE 2: Whether the T_{SSF} is used or not in this case is network operator dependent. But it must be synchronized with $T_{SCF-SSF}$ in the **SCSM.**

NOTE 3: The successful connection to the SRF causes a state transition in the **SRF FSM** from "Idle" to "Connected".

11.13.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.14   Continue procedure

11.14.1 General description

This operation is used to request the SSF to proceed with call processing at the DP at which it previously suspended call processing to await SCF instructions. The SSF continues call processing without substituting new data from the SCF.

This operation is only valid when used in a single call segment CSA and there are no more than 2 legs in the call segment. If the correct resumption of EDP-R's is required, the ContinueWithArgument operation shall be used (refer to resumption counter rules defined in clause 8.2 (IN-SSM)).

11.14.1.1   Parameters

11.14.1.1.1   Argument Parameters

None.

11.14.2 Invoking entity (SCF)

11.14.2.1   Normal procedure

SCF Preconditions:

    (1) A relationship exists between the SCF and the SSF.

    (2) **FSM for CS** is in the state "Preparing CS instructions" or "Suspended and User Interaction".

SCF Postcondition:

    (1) **FSM for CS** is in the state "Waiting for Notification or Request", in case monitoring was required, or in the state "Idle", in case no monitoring was required or in the state "Not Suspended and User Interaction".

The **FSM for CS** is in state "Preparing CS instructions". The "Continue" operation is invoked by a SLPI. This causes a **FSM for CS** transition to state "Idle" if no subsequent monitoring is required. However, if monitoring is required, like in the case of armed EDPs or outstanding report requests, the **FSM for CS** transitions to state "Waiting for Notification or Request" or "Not Suspended and User Interaction".

11.14.2.2   Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.14.3 Responding entity (SSF)

11.14.3.1 Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) BCSM: Basic call processing has been suspended at any DP.

(3) The **FSM for CS** is either in the state "Waiting for Instructions" or in the state "Waiting for End of User Interaction (WFI)" or in the state "Waiting for End of Temporary Connection (WFI)", while being suspended at the Answer DP or Mid-call DP or Suspend/Re-answer DP.

NOTE 1: The only applicable SCF-SRF user interaction operation is in this case the PlayAnnouncement.

(4) The **CSCV** is in a valid connection view state according to the CSCV transition table "Transition of DP Event to CSCV state" as described in clause 6.

NOTE 2: The Continue operation is not allowed for a single call segment CSA with more than two legs or a multi call segment CSA. In this case an error procedure is invoked.

SSF Postconditions:

(1) BCSM: Basic call processing continues, if all required resumption within the involved CS have been received, otherwise the only action is to decrement the resumption counter(s). For details refer to resumption counter in rules defined in clause 8.2 (IN-SSM).

(2) The **FSM for CS** remains in the same state if there are any pending resumptions; or
The **FSM for CS** moves to the state "Monitoring", because at least one EDP was armed, or a "CallInformationReport" or "ApplyChargingReport" was requested; and no user interaction is ongoing; or
The **FSM for CS** moves to the state "Idle", because no EDPs were armed and neither the "CallInformationReport" nor the "ApplyChargingReport" was requested.

(3) If in state "Waiting for End of User Interaction (WFI)" the **FSM for CS** moves to the state "Waiting for End of User Interaction (MON)" if any armed EDP or pending report ("CallInformationReport", "ApplyChargingReport") that is a persistent relationship.
If necessary (for CAMEL mandatory),, a guard timer T_{SSF} is set; or

moves to the state "Idle", if no armed EDP or pending reports (i.e. the relationship is ended). All IN resources including connection to SRF are released as the **FSM for CS** moves to the state "Idle".

(4) If in state "Waiting for End of Temporary Connection (WFI)", the **FSM for CS** moves to the state "Waiting for End of Temporary Connection (MON)" if there are any armed EDP or pending report (i.e. a persistent relationship). If necessary (for CAMEL mandatory), a guard timer T_{SSF} is set; or

moves to the state "Idle", if there are no armed EDP or pending reports ("CallInformationReport", "ApplyChargingReport"), that is the relationship is ended. All IN resources including connection to SRF are released as the **FSM for CS** moves to the state "Idle".

(5) Refer to clause "Call Segment Connection View (CSCV) State Description" within clause 6 for details regarding any **CSCV** state transition upon resumption due to Continue operation of the basic call process from the actual DP and CSCV state.

11.14.3.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.15 ContinueWithArgument procedure

11.15.1 General description

This operation is used to request the SSF to proceed with call processing at the DP at which it previously suspended call processing to await SCF instructions. It is also used to provide additional service related information to a User (Called Party or Calling Party) whilst the call processing proceeds.

This operation has to be send for each BCSM instance in a CS for which call processing has been suspended (indicated by legID) in case the call suspension is due to a signalling event, i.e. once for each reported intercepted event to the SCF. If call processing suspension is caused by the SCF sending a CPH operation all BCSM instances in the involved CSs will have been suspended and this operation has to be sent once for each involved CS to resume call processing (indicated by CSID). For details refer to resumption counter rules defined in clause 8.2 (IN-SSM).

For CAMEL: In general all parameters which are provided in a ContinueWithArgument operation to the gsmSSF shall replace the corresponding signalling parameter in the CCF and shall be used for subsequent call processing. Parameters which are not provided by the ContinueWithArgument operation shall retain their value (if already assigned) in the CCF for subsequent call processing.

11.15.1.1 Parameters

11.15.1.1.1 Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12:

- legorCSID;
  This parameter comprises the following alternatives:

  - legID; or

  - callSegmentID;

- alertingPattern;

- genericName;

- iNServiceCompatibilityResponse;

- forwardGVNS;

- backwardGVNS;

- extensions;

- serviceInteractionIndicatorsTwo;

- sDSSinformation;

- iSDNAccessRelatedInformation;

- originalCalledPartyID;

- callingPartyNumber;

- callingPartysCategory;

- redirectingPartyID;

- redirectionInformation;

- forwardCallIndicators;

- genericNumbers;

- cug-Interlock;

- cug-OutgoingAccess;

- chargeNumber;

- carrier;

- na-OliInfo;
  Support for CAMEL;

- suppressionOfAnnouncement;
  Support for CAMEL.

11.15.2 Invoking entity (SCF)

11.15.2.1 Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) **FSM for CS** is in the state "Preparing CS instructions" or in case of user interaction in the state "Suspended and User Interaction".

SCF Postcondition:

(1) **FSM for CS** is in the state "Waiting for Notification or Request", in case monitoring was required and no user interaction, or in the state "Idle", in case no monitoring was required. **FSM for CS** is in the state "Not Suspended and User Interaction" in case user interaction in monitoring state.

The **FSM for CS** is in state "Preparing CS instructions" or "Suspended and User Interaction". The "ContinueWithArgument" operation is invoked by a SLPI. This causes a **FSM for CS** transition to state "Idle" if no subsequent monitoring is required. However, if monitoring is required, like in the case of armed EDPs or outstanding report requests, the **FSM for CS** transitions to state "Waiting for Notification or Request" or "Not Suspended and User Interaction".

11.15.2.2 Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.15.3 Responding entity (SSF)

11.15.3.1 Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) BCSM: Basic call processing has been suspended at any DP.
For CAMEL only the DPs Collected_Info, Analysed_Info and Terminating_Attempt_Authorized are allowed.

(3) The **FSM for CS** is either in the state "Waiting for Instructions"; or
(not for CAMEL) in the state "Waiting for End of User Interaction (WFI)"; or
(not for CAMEL) in the state "Waiting for End of Temporary Connection(WFI), while being suspended at the Answer DP or Mid-call DP or Suspend/Re-Answer DP.

NOTE 1: The applicable SCF-SRF user interaction operation is PlayAnnouncement.

(4) The **CSCV** is in a valid connection view state according to the CSCV transition table "Transition of DP Event to CSCV state" as found within clause 6.

SSF Postconditions:

(1) BCSM: Basic call processing continues, if all required resumptions within the involved CS has been received, otherwise the only action is to decrement the resumption counter(s). For details refer to the definition of resumption counter riles in clause 8.2 (IN-SSM).

(2) The **FSM for CS** remains in the same state if there are any pending resumptions; or
The **FSM for CS** moves to the state "Monitoring", because no user interaction is ongoing and at least one EDP was armed, or a "CallInformationReport" or "ApplyChargingReport" was requested; or
FSM for CS moves to the state "Idle", because no EDPs were armed and neither the "CallInformationReport" nor the "ApplyChargingReport" was requested.

(3) If in state "Waiting for End of User Interaction (WFI)" the **FSM for CS** moves to the state "Waiting for End of User Interaction (MON)" if any armed EDP or pending report ("CallInformationReport", "ApplyChargingReport").that is a persistent relationship.
If necessary (for CAMEL mandatory), a guard timer T_{SSF} is set; or
moves to the state "Idle", if no armed EDP or pending reports (i.e. the relationship is ended). All IN resources including connection to SRF are released as the **FSM for CS** moves to the state "Idle".

(4) If in state "Waiting for End of Temporary Connection (WFI)", the **FSM for CS** moves to the state "Waiting for End of Temporary Connection (MON)" if there are any armed EDP or pending report (i.e. a persistent relationship). If necessary (for CAMEL mandatory), a guard timer T_{SSF} is set; or
moves to the state "Idle", if there are no armed EDP or pending reports ("CallInformationReport", "ApplyChargingReport"), that is. the relationship is ended. All IN resources including connection to SRF are released as the **FSM for CS** moves to the state "Idle".

(5) Refer to clause "Call Segment Connection View (CSCV) State Description" within clause 6 for a details regarding a possible **CSCV** state transition upon resumption due to ContinueWithArgument operation of the basic call process from the actual DP and CSCV state.

NOTE 2: The inclusion of carrier and chargeNumber parameters is only allowed for CAMEL.

11.15.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.16   CreateCallSegmentAssociation procedure

11.16.1 General description

This operation creates a new CSA. The new CSA will not contain any Call Segments after creation. The SSF is responsible for specifying a new CSA identifier for the created CSA which is unique within the SSF.

11.16.1.1   Parameters

11.16.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- extensions.

11.16.1.1.2   Result Parameters

The operation result consists of the following parameters. These parameters are defined in clause 12.

- newCallSegmentAssociation;

- extensions.

11.16.2 Invoking entity (SCF)

11.16.2.1 Normal procedure

SCF Preconditions:

(1) An SLPI has been invoked.

(2) An SLPI has determined that a 'CreateCallSegmentAssociation' operation should be sent by the SCF.

(3) An instance of the **FSM for CSA** is created by **SCME-Control**. The **FSM for CSA** is in state "SSF Control Idle".

SCF Postconditions:

(1) A relationship is established between the SCF and SSF.

(2) The **FSM for CSA** is in state "Preparing SSF Instructions".

(3) SLPI execution continues.

11.16.2.2 Error handling

Generic error handling for the operation related errors is described in clause 13 and the TCAP services used for reporting operating errors are described in clause 15.

11.16.3 Responding entity (SSF)

11.16.3.1 Normal procedure

SSF Precondition:

(1) An instance of the **FSM for CSA** is created by SSME-Control. The FSM for the CSA is in state 'Idle'.

SSF Postconditions:

(1) The **FSM for CSA** has moved from "Idle" state to state "Active".
The **CSA** CV is in state « Null ».

(2) A Return Result is sent to report the new CSA ID to the SCF.

11.16.3.2 Error handling

Generic error handling for the operation related errors is described in clause 13 and the TCAP services used for reporting operating errors are described in clause 15.

11.17 CreateOrRemoveTriggerData Procedure

11.17.1 General description

The SCF uses this operation to create a new trigger at a particular DP by downloading to the SSF the required triggering information (triggering criteria, service Key.). The operation also allows to remove a previous created non-active trigger.

11.17.1.1   Parameters

11.17.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- createOrRemove;
  If this parameter is omitted the default is the request to « create » a new trigger;

- dPName;

- triggerDPType;

- serviceKey;
  This parameter is mandatory for the request to « create » a new trigger;

- profile;
  This parameter is mandatory for the request to « create » a new trigger.
  It comprises the following alternatives:

  - access; or

  - group;
    This alternative identifies a facility group. It comprises the following fields:

    - trunkGroupID;
      This field identifies a trunk group;

    - privateFacility;
      This field indicates the particular group of private facilities to route a call;

    - huntGroup;
      This field contains a link to a multi-line hunt group. Its content is network operator specific;

    - routeIndex;
      This field contains a link to a specific trunk routing group. Its content is network operator specific;

- triggerData;

- defaultFaultHandling;
  It comprises the following sub-parameters:

  - action;

  - treatment;

- tDPIdentifier;
  This parameter is mandatory for the request to « remove » a trigger;

- extensions.

11.17.1.1.2   Result Parameters

The operation result consists of the following parameters. These parameters are defined in clause 12.

- triggerStatus;

- tDPIdentifier;

- registratorIdentifier;

- extensions.

11.17.2 Invoking entity (SCF)

11.17.2.1   Normal procedure

SCF Precondition:

(1) SLPI receives an indication (e.g. by the SMF) to create a new TDP and downloads to the SSF triggering information (triggering criteria, service Key, SCF address, etc.) associated with the TDP criteria. The SLPI may also receive an indication to remove an existing trigger and provides the SSF with the TriggerIdentifier for the trigger to be removed.

SCF Postcondition:

(1) **SCME** is in the state "Idle".

If createOrRemoveTriggerData has been successfully processed, the returnResult indicates the tDPIdentifier assigned to the new created or removed trigger (TDP). The SCF indicates to the initiating entity the result of the procedure.

11.17.2.2   Error handling

Generic error handling for the operation related error are described in clause 13 and the TCAP services used for reporting operation errors as described in clause 15.

11.17.3 Responding entity (SSF)

11.17.3.1   Normal procedure

SSF Precondition:

(1) None.

SSF Postconditions:

(1) **SSME-FSM** is in the state "Idle".

(2) The result or an error indication of "createOrRemoveTriggerData" is sent as ReturnResult to the initiating SCF.

If a new trigger is to be created it is checked whether the profile addressed by the operation exists. If so, the trigger is created and the trigger status is 'deactivated'.

NOTE: Activation may be performed using the ManageTriggerData operation or the SMS.

If an existing trigger is to be removed, it is checked if the triggerIdentifier is valid and if the trigger to be removed is not active (i.e. trigger status is 'deactivated'). If so, the trigger is removed.

The request to create an already existing (in)active Trigger is not an error case. It will only be indicated in the trigger status result returned that this Trigger *already exists*.
If a request is received to remove a Trigger which is not known by the SSF, it will be indicated in the trigger status result as *unknown trigger*.
No operation error is returned in that case.

11.17.3.2   Error handling

NOTE: In case of an invalid registratorIdentifier, a TaskRefused error is returned.

Generic error handling for the operation related error are described in clause 13 and the TCAP services used for reporting operation errors as described in clause 15.

11.18   DisconnectForwardConnection procedure

11.18.1 General Description

This operation is used in the following two cases:

1) To clear a connection to a SRF.

   This operation is used to explicitly disconnect a connection to a resource (SRF) established previously with a "ConnectToResource" or an "EstablishTemporaryConnection" operation. It is used for a forward disconnection from the SSF. An alternative solution is the backward disconnect from the SRF, controlled by the "DisconnectFromIPForbidden" parameter in the "PlayAnnouncement", "PromptAndCollectUserInformation, "PromptAndReceiveMessage" and "ScriptRun" operations.

2) To clear a connection to an assisting SSF.

   This operation is sent to the non-assisting SSF of a pair of SSFs involved in an assist procedure. It is used to disconnect the temporary connection between the initiating SSF and the assisting SSF, and the assisting SSF and its associated SRF.

   This operation is only valid when used in a single call segment CSA (i.e. a CSA with only one CS).

11.18.1.1   Parameters

11.18.1.1.1   Argument Parameters

None.

11.18.2 Invoking entity (SCF)

11.18.2.1   Normal procedure

SCF Preconditions:

   (1) A relationship exists between the SCF and the SSF.

   (2) An assist- or a relay procedure is in progress.

   (3) An SLPI has determined that a "DisconnectForwardConnection" operation has to be sent by the SCF.

SCF Postcondition:

   (1) SLPI execution may continue.

The "DisconnectForwardConnection" operation is used to instruct the SSF to disconnect the concerned forward connection to the assisting SSF or the physical entity containing the SRF.

In the **SCSM FSM** state "User Interaction", substate "Waiting for Response from the SRF", this operation is invoked by the SCF when the service logic determines that user interaction is finished and requests the SSF to disconnect the temporary connection to the assisting SSF or the SRF. The **SCSM FSM** then transitions to state "Preparing SSF Instructions".

11.18.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.18.3 Responding entity (SSF)

11.18.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) Basic call processing has been *suspended* at a DP, then
The **FSM for CS** in the *initiating SSF* is in the state "Waiting for End of User Interaction (WFI)" or "Waiting for End of Temporary Connection(WFI)"; or
The **Assisting/hand-off FSM** in the *assisting SSF* is in the state "Waiting for End of User Interaction".

(3) Basic call processing has *not been suspended* at a DP, then
the FSM for CS in the *initiating SSF* is in the state "Waiting for End of User Interaction (MON)"
or in the state "Waiting for End of Temporary Connection (MON)".

SSF Postconditions:

(1) The connection to the SRF or *assisting SSF* is released.

(2) The **FSM for CS** in the *initiating SSF* is in state "Waiting for Instructions" if basic call processing has been suspended at a DP, otherwise in state "Monitoring"; or
The **Assisting/Hand-off FSM** in the *assisting SSF* is in the state "Waiting for Instructions".

The receipt of "DisconnectForwardConnection" results in disconnecting the assisting SSF or the physical entity containing the SRF from the concerned call. It does not release the connection from the SSF back to the end user.

This operation is accepted in the **FSM for CS** states "Waiting for End of Temporary Connection(WFI)" or "Waiting for End of Temporary Connection(MON)" or "Waiting for End of User Interaction(WFI)" or "Waiting for End of User Interaction(MON)". On receipt of this operation in these states, the SSF must perform the following actions:

-   The initiating SSF releases the connection to the assisting SSF or the relay SRF.

-   The SSF resets T_{SSF}.

-   The **FSM for CS** goes to state "Waiting for Instructions" or "Monitoring".

NOTE: The successful disconnection to the SRF causes a state transition in the **SRF FSM** to "Idle". A current order (e.g. "PlayAnnouncement" or "PromptAndCollectUserInformation" or 'PromptAndReceiveMessage') is cancelled and any queued order ("PlayAnnouncement" or "PromptAndCollectUserInformation" or "PromptAndReceiveMessage") is discarded.

11.18.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.19   DisconnectForwardConnectionWithArgument procedure

11.19.1 General Description

The "DisconnectForwardConnectionWithArgument" operation is used to instruct the SSF to disconnect the concerned forward connection to the assisting SSF or the physical entity containing the SRF.

This operation is used in the following two cases:

1) To clear a connection to a SRF.

   This operation is used to explicitly disconnect a connection to a resource (SRF) established previously with a "ConnectToResource" or an "EstablishTemporaryConnection" operation. It is used for a forward disconnection from the SSF. An alternative solution is the backward disconnect from the SRF, controlled by the "DisconnectFromIPForbidden" parameter in e.g. the "PlayAnnouncement", "PromptAndCollectUserInformation", "PromptAndReceiveMessage" and "ScriptRun" operations.

2) To clear a connection to an assisting SSF.

   This operation is sent to the non-assisting SSF of a pair of SSFs involved in an assist procedure. It is used to disconnect the temporary connection between the initiating SSF and the assisting SSF, and the assisting SSF and its associated SRF.

11.19.1.1   Parameters

11.19.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- partyToDisconnect;

- extensions;

- uSIServiceIndicator;

- uSIInformation.

11.19.2 Invoking entity (SCF)

11.19.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) An assist- or a relay procedure is in progress.

(3) An SLPI has determined that a "DisconnectForwardConnectionWithArgument" operation has to be sent by the SCF.

SCF Postcondition:

(1) SLPI execution may continue.

In the **SCSM FSM** state "User Interaction", substate "Waiting for Response from the SRF", this operation is invoked by the SCF when the service logic determines that user interaction is finished and requests the SSF to disconnect the temporary connection to the assisting SSF or the SRF. The **SCSM FSM** then transitions to state "Preparing SSF Instructions".

11.19.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.19.3 Responding entity (SSF)

11.19.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) If basic call processing has been suspended at a DP, then:
the **FSM for CS** in the *initiating SSF* is in the state "Waiting for End of User Interaction (WFI)" or "Waiting for End of Temporary Connection (WFI)"; or
The **Assisting/hand-off FSM** in the *assisting SSF* is in the state "Waiting for End of User Interaction".

(3) If basic call processing has not been suspended at a DP, then:
the **FSM for CS** in the initiating SSF is in the state "Waiting for End of User Interaction (MON)" or "Waiting for End of Temporary Connection (MON)".

SSF Postconditions:

(1) The connection to the SRF or assisting SSF is released.

(2) The **FSM for CS** in the initiating SSF is in state "Waiting for Instructions" if basic call processing has been suspended at a DP, otherwise in state "Monitoring"; or
The **Assisting/hand-off FSM** in the assisting SSF is in the state "Waiting for Instructions".

(3) The receipt of "DisconnectForwardConnectionWithArgument" results in disconnecting the assisting SSF or the physical entity containing the SRF from the concerned call. It does not release the connection from the SSF back to the end user.

This operation is accepted in the initiating SSF in the **FSM for CS** states "Waiting for End of Temporary Connection (WFI)" or "Waiting for End of User Interaction (WFI)" or "Waiting for End of Temporary Connection (MON)" or "Waiting for End of User Interaction (MON)". On receipt of this operation in these states, the SSF must perform the following actions:

- The initiating SSF releases the connection to the assisting SSF or the relay SRF.

- The SSF resets T_{SSF}.

- The **FSM for CS** goes to state "Waiting for Instructions".

NOTE: The successful disconnection to the SRF causes a state transition in the **SRF FSM** to "Idle". A current order ("PlayAnnouncement" or "PromptAndCollectUserInformation" or "PromptAndReceiveMessage") is cancelled and any queued order (e.g. "PlayAnnouncement" or "PromptAndCollectUserInformation" or "PromptAndReceiveMessage") is discarded.

11.19.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.20   DisconnectLeg procedure

11.20.1 General description

This operation is used to request the SSF to release a specific leg associated with the call at any phase of the call and to retain any other leg not specified in the DisconnectLeg operation.

11.20.1.1   Parameters

11.20.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- legToBeReleased;

- releaseCause;

- extensions.

11.20.1.1.2   Result Parameters

None.

11.20.2 Invoking entity (SCF)

11.20.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) An SLPI has determined that a call party shall be released.

(3) The **FSM for CS** is in state C2 "Preparing CS Instructions".

SCF Postconditions:

(1) SLPI execution may continue. The **FSM for CSA** transits to "idle" on receiving the last pending report (if any), if the released leg was the last leg within the Call Segment Association.

(2) The **FSM for CS** is in state C2 "Preparing CS Instructions" if not the last leg in the CS is released, otherwise the **FSM for CS** moves to the state "Idle".

11.20.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.20.3 Responding entity (SSF)

11.20.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The leg to be released exists in one of the Call Segments.

(3) The **CSA** CV is in state « Multi-Segment » or « One-Segment ».

(4) The **FSM for CS** is in the state "Waiting for Instructions" or "Monitoring".

(5) The **CSCV** is in a valid connection view state according to the CSCV transition table 23.

SSF Postconditions:

(1) The SSF performs the call processing actions to release the indicated party.

(2) Any outstanding EDPs on that leg are disarmed, any pending reports will be sent.

(3) If the released leg is a controlling leg, the controlling leg status becomes "surrogate" in all the CS of the CSA.

(4) If the released leg was the last joined leg within the Call Segment, the **FSM for CS** for that CS returns to the "idle" state.

(5) If the released leg was not the last joined leg, the **FSM for CS** for the involved Call Segment moves or remains in the "Waiting for instructions" state. The remaining BCSM instances within the Call Segment move from the PIC to the O_/T_MidCall DP, if not already suspended at a DP. Note that no MidCall EDP will be reported for this case.

(6) The **CSA** is deleted if after the release of the leg, no CS remains in the CSA. Otherwise the CSA is in CSA CV state « Multi-Segment », or « One Segment », depending on the number of remaining CSs in the CSA.

(7) A Return Result is sent immediately after the successful change of the leg configuration is executed.

NOTE: This allows the SCF to be updated with the established connection view and to cater for possible interference problems with signalling events.

Table 23: Transition table for the CS

CSCV State (original state): ⇒ Operation: ⇓	Originating Setup	Originating 1-Party Setup	Stable 1-Party	Terminating Setup	1-Party	Stable 2-Party	Forward	Stable_Multi_Passive_ Party	Stable Multi-Party
DisconnectLeg (c)	CS instance deleted	Error (Originating 1-Party Setup)	Error (Stable 1-Party)	Error (Terminating_Setup)	CS instance deleted	Stable 1-Party	Error (Forward)	Error (Stable_Multi_ Passive_ Party)	Stable_Multi _ Passive_ Party
DisconnectLeg (p)	Error (Originating Setup)	Error (Originating 1-Party Setup)	CS instance deleted	CS instance deleted	Error (1-Party)	1-Party	CS instance deleted/Error (note 1)	Stable 1-Party /Stable_Multi_Passive_ Party (note 3)	Stable 2-Party /Stable Multi-Party (note 3)
NOTE 1: Delete CS instance if passive leg status is joined. If passive leg 'pending' then Error case. NOTE 2: Void. NOTE 3: State depends on the number of remaining legs. NOTE 4: Void.									

11.20.3.2   Error handling

NOTE: When a CPH operation is received in an inappropriate CSCV state, a TaskRefused error shall be returned.

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.21   EntityReleased procedure

11.21.1   General description

This operation is used to inform the SCP about the release of an entity (CS, BCSM) caused by exception or errors. It is sent by the **CSA FSM** if the TC dialogue has to be kept because of other existing entities (CS, BCSM) in this CSA which are not affected by this error/exception. This operation is not sent if the last CS was released.

The operation EntityReleased is not used if the release of the entity can be reported through other operations, e.g. EventReportBCSM, CallInformationReport.

11.21.1.1   Parameters

11.21.1.1.1   Argument Parameters

The operation argument consists of the following alternatives. These parameters are defined in clause 12.

- cSFailure; or

- bCSMFailure.

11.21.2   Invoking entity (SSF)

11.21.2.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) Any state except idle.

SSF Postcondition:

(1) If the released entity was a BCSM (leg) than only the appropriate resources are released.
   If the released entity was a CS the related **FSM for CS** moves to the state "Idle".

11.21.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.21.3 Responding entity (SCF)

11.21.3.1   Normal procedure

SCF Precondition:

   (1) A relationship exists between the SCF and the SSF.

SCF Postconditions:

   (1) The SCF-resources related to the released entity are released.

   (2) The SLPI is further executed.

11.21.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.22   EstablishTemporaryConnection procedure

11.22.1 General Description

This operation is used to create a connection between an initiating SSF and an assisting SSF as part of a service assist procedure. It can also be used to create a connection between a SSF and a SRF, for the case where the SRF exists in a separately addressable physical entity.

For CAMEL, the assistingSSPIPRoutingAddress shall contain routing digits, a correlationID and an scfID when a temporary connection is to be established between networks and no bilateral agreement exists between the involved network operators to transfer correlationID and SCFiD as separate parameters.

11.22.1.1   Parameters

11.22.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

   - assistingSSPIPRoutingAddress;

   - correlationID;
     Only allowed if the correlationID is not embedded in the "assistingSSPIPRoutingAddress";

   - partyToConnect;
     This parameter shall be present when applied in a multi call segment CSA.
     When not present in a single call segment CSA it implies that user interaction shall apply to the call segment, i.e. to all parties connected to the call segment.
     This parameter comprises the following alternatives:

      - legID; or

      - callSegmentID;

   - scfID;
     Only allowed if the scfID is not embedded in the "assistingSSPIPRoutingAddress";

   - extensions;

   - carrier;

   - serviceInteractionIndicators;

   - serviceInteractionIndicatorsTwo;

- na-OliInfo;
  Support for CAMEL;

- chargeNumber;
  Support for CAMEL.

11.22.2 Invoking entity (SCF)

11.22.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The service logic has determined that a connection is needed between the SSF and SRF or between the SSF and an assisting SSF.

SCF Postcondition:

(1) The **FSM for Assisting SSF** is "Waiting for Assist Request Instructions".

In the **SCSM FSM** state "Routing to Resource", this operation is invoked by the SCF when the service logic determines that an assisting SSF or a Direct SCF-SRF relation is needed. The **SCSM FSM** then transitions to state "Waiting for Assist Request Instructions".

11.22.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.22.3 Responding entity (SSF)

11.22.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The **FSM for CS** is in state "Waiting for Instructions" or in state "Monitoring".

(3) The SSF is not an assisting SSF.

SSF Postconditions:

(1) The SSF performs the call processing actions to route the call to the assisting SSF or SRF according to the "assistingSSPIPRoutingAddress" requested by the SCF.

(2) The CS waits for end of temporary connection.

(3) If in state "Waiting for Instructions" the **FSM for CS** moves to the state "Waiting for End of Temporary Connection (WFI)". If necessary (for CAMEL mandatory), T_{SSF} is set.

(4) If in state "Monitoring" the **FSM for CS** moves to the state "Waiting for End of Temporary Connection (MON)". If necessary (for CAMEL mandatory), a guard timer T_{SSF} is set.

On receipt of this operation in the **FSM for CS** state "Waiting for Instructions" or "Monitoring", the SSP has to perform the following actions:

- Reset the T_{SSF} (optional, for CAMEL mandatory).

NOTE 1: This "optional" means that the application timer T_{SSF} is optionally set. Whether it is used or not is network operator dependent. But it must be synchronized with $T_{SCF-SSF}$ in the **SCSM**.

- Route the call to assisting SSF or SRF using "assistingSSPIPRoutingAddress".

- The **FSM for CS** goes to state "Waiting for End of Temporary Connection (WFI)".

On receipt of this operation in the **FSM for CS** state "Monitoring", the SSP has to perform the following actions:

- Route the call to assisting SSF or SRF using "assistingSSPIPRoutingAddress".

- The **FSM for CS** goes to state "Waiting for End of Temporary Connection (MON)".

  NOTE 2: The inclusion of carrier and chargeNumber parameters is only allowed for CAMEL.

11.22.3.2   Error handling

Until the connection setup has been accepted (refer to ITU-T Recommendation Q.71) by the assisting SSF/SRF, all received failure indications from the network on the ETC establishment shall be reported to the SCF as ETC error ETCFailed (e.g. busy, congestion). Note that the operation timer for ETC shall be longer then the maximum allowed time for the signalling procedures to accept the connection.

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.23   EventNotificationCharging procedure

11.23.1 General description

This operation is used by the SSF to report to the SCF the occurrence of a specific charging event type as requested by the SCF using the"RequestNotificationChargingEvent" operation. The operation supports the options to cope with the interactions concerning the charging (refer to clause 16 charging scenarios).

As several charging events may occur during a connection configuration a possibility exists for the EventNotificationCharging operation to be invoked on multiple occasions. For each connection configuration EventNotificationCharging may be used several times.

11.23.1.1   Parameters

11.23.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- eventTypeCharging;

- eventSpecificInformationCharging;

- legID;
  This parameter indicates the leg on which the charging event type applies;

- extensions;

- monitorMode;
  When the "monitorMode" is "interrupted", the event is reported as a request, if the "monitorMode" is "notifyAndContinue" the event is reported as a notification. The "monitorMode" "transparent" is not applicable for the EventNotificationCharging operation;

- eventTypeTariff;

- eventSpecificInformationTariff.

11.23.2 Invoking entity (SSF)

11.23.2.1   Normal procedure

SSF Preconditions:

   (1) A relationship exist between the SCF and the SSF.

   (2) A charging event has been detected that is requested by the SCF.

SSF Postcondition:

   (1) No FSM state transition.

The **FSM for CS** is in any state except "idle". This operation is invoked if a charging event has been detected that is requested by the SCF. The detected charging event can be caused by: a) another SLPI or b) another exchange. Irrespective of the charging event cause, the SSF performs one of the following actions on occurrence of the charging event (according to the corresponding monitorMode):

Interrupted:

   Notify the SCF of the charging event using "EventNotificationCharging" operation: do not process the event, but discard it. However, call and existing charging processing will not be suspended in the SSF.

NotifyAndContinue:

   Notify the SCF of the charging event using "EventNotificationCharging", and continue processing the event or signal.

11.23.2.2   Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.23.3 Responding entity (SCF)

11.23.3.1   Normal procedure

SCF Preconditions:

   (1) A relationship exists between the SCF and the SSF.

   (2) A "RequestNotificationChargingEvent" has been sent at the request of a SLPI and the SLPI is expecting an "EventNotificationCharging" from the SSF.

SCF Postcondition:

   (1) No FSM state transition.

On receipt of this operation the SLPI which is expecting this notification can continue. If the corresponding monitor mode was set by the SLPI to Interrupted the SLPI prepares instructions for the SSF if necessary.

11.23.3.2   Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.24   EventReportBCSM procedure

11.24.1 General description

This operation is used to notify the SCF of a call related event previously requested by the SCF in an "RequestReportBCSMEvent" operation. The monitoring of more than one event could be requested with a "RequestReportBCSMEvent" operation, but each of these requested events is reported in a separate "EventReportBCSM" operation.

11.24.1.1   Parameters

11.24.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- eventTypeBCSM;

- eventSpecificInformationBCSM;

- legID;
  The "legID" parameter shall always be included for the events O_MidCall, O_Disconnect, T_MidCall and T_Disconnect;

  - **receivingSideID;**

- miscCallInfo;
  This parameter comprises the following subparameter:

  - messageType;

- extensions.

NOTE 1: For legID the SSF will use the option "ReceivingSideID" only.
    "ReceivingSideID" is a subparameter of parameter LegId.

The following values for "legID" are assumed;

NOTE 2: The IN CS-1 definition of this parameter makes assumptions regarding the allocation of LegID values. With the introduction of Call Party Handling, these assumptions are no longer appropriate. For this Capability Set the leg numbering is based on the following principles:

- legID = 1 is the controlling leg and legID = 2 is the passive leg in case the initial call segment created was an originating call segment (CS state 'Originating setup'). Additional legs can only be created by the SCF, in which case the SCF assigns the leg numbers;

- legID = 1 is the passive leg and legID = 2 is the controlling leg (i.e. inverse to the above) in case the initial call segment created was a terminating call segment (CS state 'Terminating setup'). Additional legs can only be created by the SCF, in which case the SCF assigns the leg numbers. For IN CS-1 implementations in the case of a mid call trigger it was assumed that legID = 2 was assigned to the party not causing the trigger and legID = 1 was assigned to the party causing the trigger.

NOTE 3: If legID not included, the following defaults are assumed for CAMEL:

- "legID" = 1 for the events O-Abandon and T-Abandon;

- "legID" = 2 for the events RouteSelectFailure, O-Busy, O-NoAnswer, O-Answer, T-Busy, T-NoAnswer, and T-Answer.

11.24.2 Invoking entity (SSF)

11.24.2.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SSF and the SCF.

(2) The **FSM for CS** is in the state "Monitoring", or in a User Interaction monitoring state
(WfEoUI(MON)/WfEoTC(MON)); or
the **FSM for CS** may be in any state, except Idle if the O/T_Abandon DP, O_Term_Seized DP, O/T_Answer
DP, O/T_Disconnect DP -or O/T_MidCall DP (when requested as "inAnyState") is armed and encountered.

NOTE: This ensures that the SCF will receive the events in the correct order. For example in case the **FSM for
CS** is in "Waiting for Instructions" state due to an O-MidCall EDP-R (SendCall) being reported and in
this state O-Term seized and O-Answer events are detected followed by the O_Disconnect event.

(3) The BCSM proceeds to an EDP that is armed.

SSF Postconditions:

(1) The **FSM for CS** stays in the same state if the message type was notification (EDP-N) and there are still EDPs
armed or a "CallInformationReport" or "ApplyChargingReport" requested.

(2) The **FSM for CS** moves to the state "idle" if the message type was notification (EDP-N) and there are no more
EDPs armed, no "CallInformationReport" or "ApplyChargingReport" are requested. If this was the last CS
within the CSA, also the **FSM for CSA** returns to idle.

(3) If the message type was request (EDP-R) the **FSM for CS** moves to the state "Waiting for Instructions" if the
FSM for CS was in the state "Monitoring". If user interaction is ongoing the **FSM for CS** moves to a User
Interaction waiting for instructions state (WfEoUI(WFI)/WfEoTC(WFI)). Call processing is interrupted.

11.24.2.2   Error handling

In case the message type is request, on expiration of T_{SSF} before receiving any operation, the SSF (**CSA FSM**) informs
the SCF about the release of the CS in the EntityReleased operation if the CS is not the last CS within the CSA,
otherwise it aborts the interaction with the SCF. The released call (i.e. call segment) is given final treatment, e.g. a final
announcement.

In case the dialogue has been locally aborted, note that TCAP will send back an abort message when the response from
the SCF will be received, as the transaction will no longer exist.

Operation related error handling is not applicable, due to class 4 operation.

11.24.3 Responding entity (SCF)

11.24.3.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SSF and the SCF.

(2) The **FSM for CS** is in the state "Preparing CS Instructions", substate "Waiting for Notification or Request".

SCF Postconditions:

(1) The **FSM for CS** stays in the substate "Waiting for Notification or Request" if the message type was notification
and there are still EDPs armed or a "CallInformationReport" or "ApplyChargingReport" requested; or
The **FSM for CS** moves to the state "Idle" if the message type was notification and there are no more EDPs
armed, no "CallInformationReport" or "ApplyChargingReport" are requested; or
The **FSM for CS** moves to the state "Preparing CS Instructions" if the message type was request.

(2) The event is reported to a SLPI, based on the dialogue ID. The SCF will prepare SSF or SRF instructions in accordance with the SLPI.

11.24.3.2   Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.25   FurnishChargingInformation procedure

11.25.1   General description

This operation is used to request the SSF to generate a logical call data record for either a specified leg or connection to a User Interaction, or to include some information in the logical call data record. The logical call data record can be mapped to either a PSTN default record or a specific IN call data record. The mapping of the logical call data record to either a PSTN default record or an IN call data record is network operator specific. The generated logical call data record is intended for off-line charging of the call. A possibility exists for the FurnishChargingInformation (FCI) operation to be invoked on multiple occasions. FCI can be applied during setup of a leg or at User Interaction in order to request to start logical call data record generation. In addition FCI can e.g. be applied during the leg or User Interaction is connected, or immediately after the leg or User Interaction is released (e.g. for follow-on calls). In this case the FCI can be used to include charge related information into the logical call data record which was started at the beginning of the call.

Furthermore, this operation allows to activate and influence charging in the SSF. The SSF is in this case the charge generation point, where the charge determination point either can be the SCF or a succeeding exchange.

The charging scenario B is supported by this operation. Refer to clause 16 "Charging scenarios supported by Core INAP".

11.25.1.1   Parameters

11.25.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- fCIBillingChargingCharacteristics.

    This parameter comprises the following alternatives:

    - fCIBCCcs1; or

    - fCIBCCsequence;
      The content of this subparameter consists of:

        - fCIBCC;

        - chargeMessageFromSCF;

    NOTE: Following sub-parameters are not relevant:
             - sub-parameters subscriberCharge and delayUntilStart within parameter chargingControlIndicators;
             - Parameters originationIdentification and destinationIdentification.

        - tariffFromSuccExchange;

        - freeFormatData;

        - chargingAddress;

        - partyToChargeIdentifier;

        - iNRecordIdentifier;

        - extensions.

11.25.2 Invoking entity (SCF)

11.25.2.1 Normal procedure

SCF Preconditions:

(1) A relationship exist between the SCF and the SSF.

(2) An SLPI has determined that a "FurnishChargingInformation" has to be sent by the SCF.

SCF Postconditions:

(1) No FSM state transition.

(2) SLPI execution may continue.

The SCSM FSM is in state "Preparing SSF instruction", "Waiting For Notification or Request", or is in state "Queuing FSM". This operation is invoked by the SCF if a SLPI results in the request of creating a logical call data record to the SSF or to include some billing or charging information into the logical call data record. In the case of call queuing, this operation may contain information pertaining to the initiation of queuing or the call queuing time duration for call logging purpose. This causes no SCSM FSM state transition. Furthermore, this operation is used by the service logic to activate/influence charging in the SSF, which then will act as the charge generation point. The service logic will then determine whether the SCF or a succeeding exchange will act a charge determination point.

If the SSF is used as the charge generation point, the charge registration point can either be in the SCF or in the SSF. If the SCF is the charge registration point the AC/ACR operation can be used by the service logic to get information about the charging. If the SSF is the charge registration point, the FCI is used by the service logic to provide specific service related data for the logical call data record.

11.25.2.2 Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.25.3 Responding entity (SSF)

11.25.3.1 Normal procedure

SSF Preconditions:

(1) A relationship exist between the SCF and the SSF.

(2) **FSM for CS** State "Waiting for Instructions"; or

FSM for CS State "Waiting for End of User Interaction(WFI)"; or

FSM for CS State "Waiting for End of User Interaction (MON)"; or

FSM for CS State "Waiting for End of Temporary Connection(WFI)"; or

FSM for CS State "Waiting for End of Temporary Connection(MON)"; or

FSM for CS State "Monitoring"; or

Assisting/hand-off -FSM in the State "Waiting for Instructions".

(3) The FCI operation is allowed for both controlling and passive legs (leg status: pending or joined) as well as an SRF connection as indicated by the chargingAddress.

SSF Postconditions:

(1) No FSM state transition.

(2) If the logical call data record is associated to a leg then the record is closed when the leg is released. If the logical call data record is related to a call segment then the record is closed on completion of the User Interaction, i.e. when SRF connection is released according to the applied SRF control procedure (e.g. SSF relay, Assist, Hand-off method).

For CAMEL, if an FCI operation is received for the called party when the gsmSSF is in state 'Monitoring', or is suspended in one of the following DPs then the charging information shall be included in the logical call record for the leg that has been or is to be established:

- Collected_Info;

- O_Answer;

- Terminating_Attempt_Authorized; or

- T_Answer.

For CAMEL, if an FCI operation is received for the called party when the gsmSSF is suspended in any other DP then the charging information shall be included in the logical call record created for the last failed or disconnected called party.

On receipt of this operation the SSF perform actions according to the offline charging scenario which is applicable using the information elements included in the operation.
The following actions can be performed:

- Generation of a logical call data record.

- Include the information received in the operation in the logical call data record.

- Activate and influence charging in the SSF.

The SSP records charge related data like for example the call duration, begin time stamp or end time stamp.

When the FCI operation, with legID, is received as a response upon a sent operation (e.g. ERB, IDP) reporting the DP encountered related to a release event from the called destination (e.g. 'routeSelectFailure', 'busy', 'no-Answer' or 'calledPartyDisconnect') then two cases exist:

1) The iNCallRecordType indicates either Overwrite (OverWriteDataForCallRecord) or Append (AppendDataToCallRecord) data to the logical call data record. In this case the FCI operation is related to the leg being released.

2) The iNCallRecordType indicates either Generate (GenerateCallRecord) or no data for (NoCallRecordInformation) logical call data record. In this case the FCI operation is related to the leg that is expected to be created at a subsequent call setup.

CPH operations besides from releasing a leg shall not result in any change of the SSF charge process and the logical call data record applied to a leg, e.g. in case a leg is moved from one call segment into another.

In case the parameter chargeMessageFromSCF is received from the SCF then the provided tariff or add on charge shall be used to influence the charging in the SSF. The reception of this parameter in the SSF will not cause any signalling in the network.

To indicate that the SSF must charge according to the information received from a succeeding exchange the tariffFromSuccExchange parameter is set to either:

1) takeIntoAccountNoAOC
At the reception of the charge message in the SSF the message is acknowledged by the SSF if it contains either ChargingTariffInformation or AddOnChargingInformation for subscriber charging. The charging message is in this case not further propagated in the network. In case further charge messages are received these will be applied for the charging in the SSF (in case a charging message contains a new tariff then the new tariff will replace the existing tariff when the new tariff becomes valid). The further charging messages will also be stopped and then acknowledged by the SSF.

2) takeIntoAccountTranslateIntoAOC
At the reception of a charging message in the SSF that contains either ChargingTariffInformation or AddOnChargingInformation for subscriber charging the SSF will act as both charge registration and charge generation point and must act as described in ES 201 296. However, the SSP is also the connection control point and the tariffs received from succeeding exchange must be started and stopped as described in ES 201 296. In case further charging messages are received these will be applied for the charging in the SSF (in case a charging message contains a new tariff then the new tariff will replace the existing tariff when the new tariff becomes valid).

For interworking with the SendChargingInformation operation with regard to the parameter 'tariffInfoFromSuccExchangeSCI' see table 44 in clause 16.

Note, that there are the following restrictions with respect to the forwarding of charge messages received from B-side:

- In case of forwarding charge messages from more than one outgoing leg to a special A-party the rules at the charge generation or registration point of the underlying basic network regarding multiple tariff determination have to be regarded. Dependent on the 'network identification' of the sender of the charge messages they can overwrite each other. According to the ES 201 296 no multiple tariff determination for one network operator is foreseen.

- It is only possible to forward charge messages from a special destination to the A party, if they are connected within the same Call Segment at the connection set-up.

The tariffFromSuccExchange parameter is only allowed if the operation addresses a leg, and the leg is either a passive leg in an O_BCSM or a controlling leg in a T_BCSM (an "outgoing" leg).

Furthermore, the SCF shall assure that the usage of the tariffFromSuccExchange parameter with values "takeIntoAccountNoAOC" or "takeIntoAccountTranslateIntoAOC", is not combined with RNC/ENC monitoring for chargingTariffInfomation or addOnChargingInformation in interrupt mode.

If a charging message is received and armed in interrupt mode, it is reported to the SCF and is not processed in the SSF.

The tariff applied for SSF charging can then be determined by:

a) the SCF; or

b) the SSF; or

c) a succeeding exchange; or

d) the post processing function.

The SSF can either account the generated charge real time or off line. In both cases, the total charge must include both the charge related to the tariff and the charge related to addOnCharge.

For CAMEL, no logical call data record is output at the end of a user interaction.

11.25.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.26 InitialDP procedure

11.26.1 General description

This operation is sent by the SSF after detection of a TDP-R in the BCSM, to request the SCF for instructions to complete the call.

11.26.1.1 Parameters

11.26.1.1.1 Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- serviceKey;
- calledPartyNumber;
- callingPartyNumber;
- callingPartyBusinessGroupID;
- callingPartysCategory;
- cGEncountered;
- iPSSPCapabilities;
- iPAvailable;
- locationNumber;
- originalCalledPartyID;
- terminalType;
- extensions;
- highLayerCompatibility;
- serviceInteractionIndicators;
- additionalCallingPartyNumber;
- forwardCallIndicators;
- bearerCapability;
- eventTypeBCSM;
- redirectingPartyID;
- redirectionInformation;
- cause;
- iSDNAccessRelatedInformation;
- iNServiceCompatibilityIndication;
- genericNumbers;
- serviceInteractionIndicatorsTwo;
- forwardGVNS;

- createdCallSegmentAssociation;

- uSIServiceIndicator;

- uSIInformation;

- carrier;

- cCSS;

- vPNIndicator;

- cNInfo;

- callReference;

- routeingNumber;

- callingGeodeticLocation;

- globalCallReference;

- cug-Index;

- cug-Interlock;

- cug-OutgoingAccess;

- iMSI;
  Support for CAMEL;

- subscriberState;
  Support for CAMEL;

- locationInformation;
  Support for CAMEL;

- ext-basicServiceCode;
  Support for CAMEL;

- callReferenceNumber;
  Support for CAMEL;

- mscAddress;
  Support for CAMEL;

- calledPartyBCDNumber;
  Support for CAMEL;

- timeAndTimezone;
  Support for CAMEL;

- gsm-ForwardingPending;
  Support for CAMEL;

- InitialDPArgExtension;
  It consists of the following subparameters:

  - gmscAddress;
    Support for CAMEL.

11.26.2 Invoking entity (SSF)

11.26.2.1   Normal procedure

SSF Preconditions:

(1) An event fulfilling the criteria for the DP being executed has been detected.

(2) Call gapping and SS7 overload are not in effect for the call, and the call is not to be filtered.

SSF Postcondition:

(1) A relationship has been established if the DP was armed as a TDP-R. The **FSM for CS** moves to the State "Waiting for Instructions".

Following a trigger detection (due to the DP criteria assigned being met) related to an armed TDP in the BCSM caused by a call origination attempt, the SSF checks if call gapping, SS7 overload or service filtering are not in effect for the related call segment.

If these conditions are met, then the "InitialDP" operation is invoked by the SSF. The address of the SCF the "InitialDP" operation has to be sent to is determined on the base of trigger related data. The SSF provide as many parameters as available. In some cases, some parameters must be available (such as "callingPartyNumber" or "callingPartyCategory"). This is to be handled appropriately by the SSF in its trigger table (to know that such parameter are necessary for some triggering conditions) and in conducting the necessary action to get these parameters if they are not available (For instance if non-SS7 signalling is used, it may be possible to request the "callingPartyCategory" from a preceding exchange).

Otherwise, the call control is given back to the underlying network.

If the DP was armed as a TDP-R a relationship is established to the SCF. The SSF application timer T_{SSF} is set when the SSF sends "InitialDP" for requesting instructions from the SCF. It is used to prevent excessive call suspension time.

In the case that the switch (CCF) generates the globalCallReference ID parameter with callReference ID length zero and a Call Reference parameter is received from ISUP, both parameters will be passed in INAP.

11.26.2.2   Error handling

If the destination SCF is not accessible then the call is given final treatment.
In CAMEL the default call handling is used.

On expiration of T_{SSF} before receiving any operation, the SSF aborts the interaction with the SCF and the call is given final treatment, e.g. routing to a final announcement.

If the calling party abandons after the sending of "InitialDP", then the SSF aborts the relationship by means of an abort to TCAP.

In case the dialogue has been locally aborted, note that TCAP will send back an abort message when the response from the SCF will be received, as the transaction will no longer exist.
Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.26.3 Responding entity (SCF)

11.26.3.1   Normal procedure

SCF Precondition:

None.

SCF Postcondition:

(1) An SLPI has been invoked.

On receipt of "InitialDP" operation the **SCSM** moves from "Idle" to the state "Preparing SSF Instructions", a relationship to the related SSF is created. A Service Logic Program Instance (SLPI) is invoked for processing the 'InitialDP' operation based on the "serviceKey" parameter. By means of this relationship, the SCF may influence the Basic Call Processing in accordance with the service logic invoked.

The actions to be performed in the SLPI depend on the parameters conveyed via this operation and the SLPI, i.e. the requested IN service, itself.

11.26.3.2   Error handling

If the "InitialDP" operation is rejected then the **SCSM** remains in "Idle". The maintenance function is informed and no SLPI is invoked.

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.27   InitiateCallAttempt procedure

11.27.1   General description

This operation is used to request the SSF to create a new call to one call party using the address information provided by the SCF (e.g. wake-up call). An EDP-R must be armed on e.g. answer and all the call failure events, in order to have the SCF treat this call appropriately when either of these events is encountered. If the Service Logic wishes to know the created CSA-ID, e.g. to be able later on to merge CSs from other CSAs with this one, a CreateCallSegmentAssociation operation needs to proceed the InitiateCallAttempt operation.

InitiateCallAttempt can also be used to create additional Call Segments within an existing Call Segment Association.

11.27.1.1   Parameters

11.27.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- destinationRoutingAddress;

- alertingPattern;

- iSDNAccessRelatedInformation;

- extensions;

- serviceInteractionIndicators;

- callingPartyNumber;
  If this parameter is not sent by the SCF, the SSF may supply a network dependent default value;

- legToBeCreated;
  When not provided, a default LegID of 1 is assumed. In the existing 'CSA' case, this parameter shall be provided by the SCF;

- newCallSegment;
  When not provided, a default CSID of 1 is assumed ('new CSA' case). In the existing CSA case, this parameter shall be provided by the SCF;

- iNServiceCompatibilityResponse;

- serviceInteractionIndicatorsTwo;

- carrier;

- correlationID;
  This parameter is only allowed if the correlationID and scfID are not embedded in the "destinationRoutingAddress";

- scfID;
  This parameter is only allowed if the correlationID and scfID are not embedded in the "destinationRoutingAddress";

- callReference;

- calledDirectoryNumber;

- originalCalledPartyID;

- callingPartysCategory;

- redirectingPartyID;

- redirectionInformation;

- displayInformation;

- forwardCallIndicators;

- genericNumbers;

- forwardGVNS;

- bearerCapability;

- globalCallReference;

- cug-Interlock;

- cug-OutgoingAccess.

11.27.2 Invoking entity (SCF)

11.27.2.1   Normal procedure

SCF Preconditions:

(1) An SLPI has been invoked.

(2) An SLPI has determined that an "InitiateCallAttempt" operation should be sent by the SCF.

(3) The **FSM for CSA** is "Preparing SSF Instructions" (existing relationship) or is in state "SSF Control Idle" (new relationship).

SCF Postconditions:

(1) A relationship is established between the SCF and SSF.

(2) The **FSM for CS** is in state "Preparing CS Instructions".

(3) SLPI execution continues.

The **FSM for CS** moves to state "Preparing SSF Instructions" when the service logic invokes this operation. In order to enable the establishment of a relationship between the SCF and SSF and to allow the SCF to control the created call appropriately, the SLPI shall monitor for the BCSM event(s) which report the result of the created call setup. This includes DP Analysed_Information or DP Route_Select_Failure, DP Authorize_Route_Failure, DP O_Called_Party_Busy, DP O_NO-Answer, and DP O_Answer. Any other Non-Call_Processing_Instructions may be sent as well.

The "InitiateCallAttempt" operation creates a BCSM instance in the SSF but the SSF suspends the call processing of this BCSM. The SLPI shall send a "Continue" or "ContinueWithArgument" operation to request the SSF to route the call to the specified destination. The **FSM for CS** shall proceed as specified at the procedure for the "Continue" respective "ContinueWithArgument" operation.

When the above described procedure shall be part of the establishment of the relationship, then the. operations up to and including the "Continue" or "ContinueWithArgument" operation shall be sent together in the same message (TC-BEGIN request primitive) to the SSF unless the CreateCallSegmentAssociation operation proceeds the InitiateCallAttempt.

The SCF shall start a response Timer T_{SCF} when the "InitiateCallAttempt" operation is sent. The response Timer shall supervise the confirmation of the dialogue from SSF, the value of T_{SCF} shall be equal or less than the network no answer timer.

11.27.2.2 Error handling

On expiration of T_{SCF} the SCF shall abort the dialogue, report the abort to the maintenance functions and inform the SLPI on the failure of dialogue establishment. The **FSM for CS** moves to the Idle state.

Generic error handling for the operation related errors are described in clause 13, the TCAP services used for reporting operation errors are described in clause 15.

11.27.3 Responding entity (SSF)

11.27.3.1 Normal procedure

SSF Preconditions:

(1) The **FSM for the CSA** is in state "Idle" or "Active".

(2) The **CSCV** is in a valid connection view state according to the CSCV transition table 24.

SSF Postconditions:

(1) A new originating BCSM has been created, call processing is suspended at DP Origination_Attempt_Authorized.

(2) The **FSM for CS** has moved from state "Idle" to state "Waiting for Instructions".

Table 24: Transition for New CS

CSCV State (original state): ⇒ Operation: ⇓	Null
InitiateCallAttempt	Originating 1-Party Setup (see note)
NOTE: In case of operation error handling, no new CS is created.	

Upon reception of "InitiateCallAttempt", the SSF creates a new originating BCSM and suspends the call processing of this BCSM at DP Origination_Attempt_Authorized. All subsequent operations are treated according to their normal procedures.

The properties and capabilities normally received from or associated to the calling party, required for the call setup shall have a network dependent default value. If a calling party number is supplied by the SCF, these properties may be dependent on the received calling party number.

11.27.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13, the TCAP services used for reporting operation errors are described in clause 15.

11.28   ManageTriggerData Procedure

11.28.1 General description

This operation is used to activate, deactivate or retrieve the status of one or several trigger detection point linked to a profile known at a switch, e.g. related to an access line. This operation is used for service logic controlled IN management purposes. The status or a success indication is sent to the SCF as Return Result of this operation.

11.28.1.1   Parameters

11.28.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- actionIndicator;

- triggerDataIdentifier;
  It comprises the following alternatives:

  - profileAndDP; or

  - profile;

- registratorIdentifier;

- extensions;

- tDPIdentifier;
  If this parameter is omitted all triggers associated with the DP are considered.

11.28.1.1.2   Result Parameters

The operation result consists of the following alternatives. These parameters are defined in clause 12.

- oneTriggerResult; or

- severalTriggerResults.

11.28.2 Invoking Entity (SCF)

11.28.2.1   Normal Procedure

SCF Precondition:

(1) SLPI detects that trigger data managing actions are to be performed.

SCF Postcondition:

(1) **SCME** is in the state "Idle".

If the service logic at the SCF requests the activation, deactivation or status interrogation of trigger data at the SSF the **SCF-FSM** sends a ManageTriggerData operation with corresponding ActionIndicator.

If ManageTriggerData has been successfully processed the ReturnResult indicates the action performed or in case of interrogation the state of the TDP.

11.28.2.2　　Error Handling

Generic error handling for the operation related error are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.28.3　Responding Entity (SSF)

11.28.3.1　　Normal Procedure

SSF Precondition:

　(1) None.

SSF Postconditions:

　(1) **SSME-FSM** is in the state "Idle".

　(2) If ManageTriggerData has been successfully processed the ReturnResult indicates the action performed or in case of interrogation the state of the TDP.

It is checked whether corresponding TDP and subscriber profile addressed by operation ManageTriggerData exist. If so, the TDP is activated, deactivated or the status is retrieved. The result or an error indication is sent back as ReturnResult of ManageTriggerData to the initiating SCF.

The (de)activation of an already (in)active TDP is not an error case. It will only be indicated that this TDP was already (in)active so that the SCF may provide an appropriate indication to the requesting entity, (e.g. an SMF).

If one or more of the TDP of the list is not known by the SSF, it will be indicated in the ActionPerformed result for that TDP, that this TDP is unknown. No operation error is returned in that case.

Except for the cases listed above, if an error occurs, an error indication is sent back to the SCF. In such a case the status of each of TDPs is unchanged as if the operation has not been received.

11.28.3.2　　Error Handling

Generic error handling for the operation related error are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.29　MergeCallSegments procedure

11.29.1　General description

This operation is used to request the SSF to merge two associated CSs into a single CS. It (re)establishes the bearer connection between all involved legs. This operation is the inverse of the SplitLeg operation regarding a single leg.

In merging the specified "source" CS, the conditions of the leg(s) which the CS has: the armed EDPs, the ApplyChargingReport pending, the EventNotificationCharging pending, and the CallInformationReport pending, are also applied for the same leg(s) after being merged.

11.29.1.1   Parameters

11.29.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- sourceCallSegment;

- targetCallSegment;
  When not specified, the source CS shall be merged with the initial CS;

- extensions.

11.29.1.1.2   Result Parameters

None.

11.29.2 Invoking entity (SCF)

11.29.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) An SLPI has determined that two call segments shall be merged into a single call segment.

(3) The **FSM for CS** is in state C2 "Preparing CS Instructions".

SCF Postconditions:

(1) SLPI execution may continue.

(2) The **FSM for CS** remains in state C2 "Preparing CS Instructions".

11.29.2.2   Error handling

Generic error handling for the operation related error are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.29.3 Responding entity (SSF)

11.29.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The **CSA** CV is in state "Multi-Segment".

(3) The **FSM for CS** of the source CS and target CS are in the state "Waiting For Instructions" or "Monitoring".

(4) The **CSCV** is in a valid connection view state according to the **CSCV** transition tables below.

SSF Postconditions:

(1) The **FSM for CS** of the source CS returns to "idle" state.

(2) The **FSM for CS** for the target Call Segment moves or remains in the "Waiting for instructions" state. The associated BCSM instances within the target Call Segment moves over from the PIC to the DP of the corresponding O_/T_MidCall DP, when not already suspended at a DP. Note that no MidCall EDP will be reported for this case.

(3) While the **CSA** CV is in state "Multi-Segment" a transition shall occur to the **CSA** CV state "One Segment", if after merging of call segments the **CSA** hereafter contains only one call segment.

(4) A Return Result is sent immediately after the successful change of the leg(s) configuration is executed.

NOTE: This allows the SCF to be updated with the established connection view and to cater for possible interference problems with signalling events.

Table 25: Transition table for all source CS

CSCV State(source CS original state): ⇒ Operation: ⇓	All states
MergeCallSegments	CS instance deleted (see note)
NOTE: The **FSM for CS** of the merged source CS is deleted for the source CS in case the SSF preconditions are met for MergeCallSegments. In case of operation error handling, the **FSM for CS** remains in the same CSCV state.	

Table 26: Transition for Target CS

CSCV State (source CS original state): ⇒ Operation (target CS original state): ⇓	Originating Setup	Originating 1 Party Setup	Stable 1-Party	Terminating Setup	1-Party
MergeCallSegments Originating_Setup	Error (Originating Setup)	Error (Originating 1 Party Setup)	Stable 2-Party	Error (Terminating Setup)	Error (1-Party)
MergeCallSegments Orig._1 Party Setup	Error (Originating Setup)	Error (Originating 1 Party Setup)	Error (Stable 1-Party)	Error (Terminating Setup)	Error (1-Party)
MergeCallSegments Stable 1-Party	Stable 2-Party note 3	Error (Originating 1 Party Setup)	Stable_Multi_ Passive_Party	Error (Terminating Setup) or Stable_Multi_ Passive_Party note 3	Stable 2-Party
MergeCallSegments Terminating-Setup	Error (Originating Setup)	Error (Originating 1 Party Setup)	Error (Stable 1-Party)	Error (Terminating Setup)	Error (1-Party)
MergeCallSegments 1_Party	Error (Originating Setup)	Error (Originating 1 Party Setup)	Stable 2-Party	Error (Terminating Setup) or Stable_2_Party notes 3, 4	Error (1-Party)
MergeCallSegments Stable 2-Party	Error (Originating Setup)	Error (Originating 1 Party Setup)	Stable Multi-Party	Error (Terminating Setup)	Error (1-Party)
MergeCallSegments Forward	Error (Originating Setup)	Error (Originating 1 Party Setup)	Error (Stable 1-Party)	Error (Terminating Setup)	Error (1-Party)
MergeCallSegments Stable_Multi_ Passive_Party	Stable Multi-Party note 3	Error (Originating 1 Party Setup)	Stable_Multi_ Passive_Party	Error (Terminating Setup)	Stable Multi-party
MergeCallSegments Stable Multi-Party	Error (Originating Setup)	Error (Originating 1 Party Setup)	Stable Multi-party	Error (Terminating Setup)	Error (1-Party)

Table 26a: Transition for Target CS

CSCV State (source CS original state): ⇒ Operation (target CS original state): ⇓	Stable_2_Party	Forward	Stable_Multi_Passive_Party	Stable_Multi_Party
MergeCallSegments O_Setup	Error (Stable 2-Party)	Error (Forward)	Stable_Multi_Party	Error (Stable Multi-Party)
MergeCallSegments O_1 Party Setup	Error (Stable 2-Party)	Error (Forward)	Error (Stable_Multi_ Passive_Party)	Error (Stable Multi-Party)
MergeCallSegments Stable 1-Party	Stable_Multi_Party	Error (Forward)	Stable_Multi_Passive_Party	Stable Multi-Party
MergeCallSegments T-Setup	Error (Stable 2-Party)	Error (Forward)	Error (Stable_Multi_ Passive_Party)	Error (Stable Multi-Party)
MergeCallSegments 1_Party	Error (Stable 2-Party)	Error (Forward)	Stable Multi_Party	Error (Stable Multi-Party)
MergeCallSegments Stable 2-Party	Error (Stable 2-Party)	Error (Forward)	Stable_Multi-Party	Error (Stable Multi-Party)
MergeCallSegments Forward	Error (Stable 2-Party)	Error (Forward)	Error (Stable_Multi_Passive_ Party)	Error (Stable_Multi_Party)
MergeCallSegments Stable_Multi_ Passive_Party	Stable_ Multi-Party	Error (Forward)	Stable_Multi_Passive_ Party	Stable Multi-Party
MergeCallSegments Stable M-Party	Error (Stable 2-Party)	Error (Forward)	Stable_ Multi-party	Error (Stable Multi-Party)

| NOTE 1: Void. |
| NOTE 2: Void. |
| NOTE 3: For the source CSCV state BCSM is suspended at any valid DP for this CSCV state, else an error case. |
| NOTE 4: The controlling leg becomes associated to a BCSM whose type is not necessary compatible with that indicated by the BCSM type attribute. |

11.29.3.2 Error handling

NOTE: When a CPH operation is received in an inappropriate CSCV state, a TaskRefused error shall be returned.

Generic error handling for the operation related error are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.30 MoveCallSegments procedure

11.30.1 General description

This operation moves one or more Call Segment(s) from the source Call Segment Association to the target Call Segment Association. The SCF specifies a new identifier for the moved CS, as well as for each leg associated with the moved CS.

This operation ends the association between the moved Call Segment and any Call Segment(s) remaining in the source Call Segment Association.

A moved CS inherits the TCAP transaction opened for the target CSA.

A Call Segment Association may contain any number of Call Segments. The number of Call Segments that may be moved into or out of a CSA is not limited by physical restrictions on the number of parties a particular switch implementation can support in a multi-party call.

In moving the specified source Call Segment(s), the condition of the leg(s) related to the moved call segment(s): the armed EDPs, the ApplychargingReport pending, the EventNotificationCharging pending, and the CallInformationReport pending and the CallInformationReport pending, are also applied for the same leg(s) after being moved.

ETSI

11.30.1.1   Parameters

11.30.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- targetCallSegmentAssociation;
- callSegments;
- legs;
- extensions.

11.30.1.1.2   Result Parameters

None.

11.30.2 Invoking entity (SCF)

11.30.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The Call is in an appropriate Call Connection View (CSCV) state.

(3) The **FSM for CS** is in state C2 "Preparing CS Instructions".

(4) The SLPI is processing the incoming request.

SCF Postconditions:

(1) SLPI execution may continue.

(2) The **FSM for CS** remains in the same state C2 "Preparing CS Instructions".

11.30.2.2   Error handling

Generic error handling for the operation related errors is described in clause 13 and the TCAP services used for reporting operating errors are described in clause 15.

11.30.3 Responding entity (SSF)

11.30.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The target **CSA** CV exists in state "Multi-Segment" or "One-Segment " or "Null".

(3) The **FSM for CS** of the source CS is in the state "Waiting for Instructions" or "Monitoring".

(4) In the source CSA if the CS to be moved has a joined controlling leg, then the target CSA must not contain a joined or pending controlling leg. Exception see note 5.

(5) The **CSCV** is in a valid connection view state according to the CSCV transition tables below.

SSF Postconditions:

(1) The SSF performs the necessary actions to move the indicated Call Segment(s) into the target CSA. All legs with the status "joined" of the source CS are now connected to the Connection Point of the new CS in the target CSA.

(2) The **FSM for Call Segment** of the moved Call Segment(s) in the target **CSA** remains in the same state.

(3) Void.

(4) If the MoveCallSegments operation results in a source CSA with no remaining Call Segments, the source CSA is deleted.

(5) The **CSA** CV is in state "Multi-Segment" or "One-Segment" depending on the number of CSs in the CSA.

(6) A Return result is sent immediately after the successful change of the call segment(s) configuration is executed.

Table 27: Transition table for all moved CSs in the source CSA

CSCV State (original state): ⇒ Moved CS: CSCV State (matrix shows result) Operation: ⇓	All states
MoveCallSegments	CS instance deleted (see note)
NOTE: The **FSM for CS** of the moved CS returns to idle state in the source CSA in case the SSF preconditions are met. In case of operation error handling, the **FSM for CS** is not moved and remains on the same CSCV.	

Table 28: Target CSA: CSCV state transition for the new CS moved from the source CSA into the target CSA

Source CSA Moved CS: CSCV State (original state): ⇒ Target CSA: New CS: CSCV State (matrix shows result) Operation and precondition: ⇓	Originating Setup	Originating 1 Party Setup	Stable 1-Party	Terminating Setup	1-Party
MoveCallSegments: Target CSA contains a CS with a controlling leg	Error (Originating Setup)	Originating 1 Party Setup	Stable 1-Party	Terminating Setup notes 4, 5 or Error (Terminating Setup)	Error (1-Party)
MoveCallSegments: Target CSA contains no CS with a controlling leg	Originating_ Setup	Originating 1 Party Setup	Stable 1-Party	Terminating Setup note 4 or Error (Terminating Setup)	1-Party

Table 28a: CSCV state transition for the new CS moved from the source CSA into the target CSA

Source CSA Moved CS: CSCV State (original state): ⇒ Target CSA: New CS: CSCV State (matrix shows result) Operation and precondition: ⇓	Stable 2-Party	Forward	Stable_Multi_Passive_Party	Stable Multi-Party
MoveCallSegments: Target CSA contains a CS with a joined/pending controlling leg	Not Applicable Error (Stable 2-Party)	Forward	Stable_Multi_Passive_Party	Not Applicable Error (Stable Multi-Party)
MoveCallSegments: Target CSA contains no CS with a joined/pending controlling leg	Stable 2-Party	Forward	Stable_Multi_Passive_Party	Stable Multi-Party

Table 29: Target CSA: CSCV state transition for the other CS(s) having a relationship with the controlling leg of the imported CS (i.e. the moved CS)

Target CSA Remaining CS: CSCV Original State: ⇒ note 1 New CS: CSCV State (matrix shows result) Operation and preconditions ⇓	Originating Setup	Originating 1 Party Setup	Stable 1-Party	Terminating Setup	1-Party	Stable 2-Party	Forward	Stable_Multi_Passive_Party	Stable Multi-Party
MoveCallSegments: Imported CS contained a joined/pending controlling leg	Not Applicable	Originating 1 Party Setup	Stable 1-Party	Terminating_ Setup	Not Applicable	Not Applicable	Forward	Stable_Multi_ Passive_Party	Not Applicable
MoveCallSegments: Imported CS contained no joined/pending controlling leg	Originating_ Setup	Originating 1 Party Setup	Stable 1-Party	Terminating_ Setup	1-Party	Stable 2-Party	Forward	Stable_Multi_ Passive_Party	Stable Multi-Party
NOTE 1: The **FSM for CS** of the moved CS returns to idle state in the source CSA in case the SSF preconditions are met. In case of operation error handling, the **FSM for CS** is not moved and remains in the same CSCV state.									
NOTE 2: Only one joined controlling leg is allowed within a CSA, i.e. transitions that would imply a result with a CSA that would have two controlling legs are marked as "Not Applicable".									
NOTE 3: Void.									
NOTE 4: The T_BCSM in the moved CS is suspended at any valid DP for this CSCV state, else an error case. However, call processing cannot be resumed (i.e. using a Continue, ContinueWithArgument operation is not allowed) for the moved CS, but the passive leg in the moved CS may become connected to a joined leg in another CS (MoveLeg, MergecallSegments).									
NOTE 5: This transition provides an exception case with more than one controlling leg in the CSA. It implies the transient existence in the CSA of the controlling leg in the moved CS together with a joined/pending controlling leg in another CS.									

Table 30: Void

11.30.3.2   Error handling

NOTE: When a CPH operation is received in an inappropriate CSCV state, a TaskRefused error shall be returned.

Generic error handling for the operation related errors is described in clause 13 and the TCAP services used for reporting operating errors are described in clause 15.

11.31   MoveLeg procedure

11.31.1   General description

This operation requests the SSF to move the Leg from one source CS to another target CS with which it is associated.

The effect of MoveLeg for the controlling Leg is to interrupt the current communication of the controlling Leg, without clearing the passive Leg on that communication, and to establish communication for the controlling Leg with another passive leg in another CS with which it is associated. Only the controlling Leg is moved.

The effect of MoveLeg for the passive Leg is to move the passive Leg and associated BCSM instance from one source CS to another target CS with which it is associated.

In moving the specified leg, the conditions of the leg: the armed EDPs, the ApplyChargingReport pending, the EventNotificationCharging pending, and the CallInformationReport pending, are also applied for the same leg after being moved. This operation is the inverse of the SplitLeg operation.

11.31.1.1   Parameters

11.31.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- legIDToMove;

- targetCallSegment;

- extensions.

11.31.1.1.2   Result Parameters

None.

11.31.2   Invoking entity (SCF)

11.31.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The Call is in an appropriate Call Connection View (**CSCV**) state.

(3) The **FSM for CS** is in state C2 "Preparing CS Instructions".

(4) A relationship has been established and the SLPI is processing the incoming request.

SCF Postconditions:

(1) SLPI execution may continue.

(2) The **FSM for CS** remains in the same state C2 "Preparing CS Instructions".

11.31.2.2 Error handling

Generic error handling for the operation related errors is described in clause 13 and the TCAP services used for reporting operating errors are described in clause 15.

11.31.3 Responding entity (SSF)

11.31.3.1 Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The **CSA** CV is in state "Multi-Segment ".

(3) The leg to be moved exists in one of the call segments.

(4) The **FSM for CS** is in the state "Waiting For Instructions" or "Monitoring".

(5) The **CSCV** is in a valid connection view state according to CSCV transition tables below.

SSF Postconditions:

(1) The SSF performs the appropriate call processing actions.

(2) If the moved leg was not the last joined leg within the CS, then the **FSM for CS** for the source and target Call Segments moves or remains in the "Waiting for instructions" state. The remaining BCSM instances within the two involved Call Segments move from the PIC to the DP of the corresponding O_/T_MidCall DP, when not already suspended at a DP. Note that no MidCall EDP will be reported for this case.

(3) If the moved leg was the last joined leg in the source CS, the **FSM for CS** for the source Call Segment will move to the 'idle' state.

(4) While the **CSA** CV is in state "Multi-Segment", a transition may occur to the **CSA** CV state "One Segment ", if moving a leg causes the deletion of its source CS and the CSA hereafter contains only " One Segment".

(5) A Return Result is sent immediately after the successful change of the leg configuration is executed.

NOTE: This allows the SCF to be updated with the established connection view and to cater for possible interference problems with signalling events.

Table 31: Transition for Source CS

CSCV State (original state): ⇒ Operation: ⇓	Originating Setup	Originating 1-Party Setup	Stable 1-Party	Terminating Setup	1-Party	Stable_2_Party	Forward	Stable_Multi_Passive_Party	Stable_Multi_Party
MoveLeg (c)	CS instance deleted or Error (Originating Setup) note 3	Error (Originating 1-Party Setup)	Error (Stable 1-Party)	Error (Terminating Setup)	CS instance deleted	Stable_1_Party	Error (Forward)	Error (Stable_Multi_Passive_Party)	Stable_Multi_Passive_Party
MoveLeg (p)	Error (Originating Setup)	Error (Originating 1-Party Setup)	CS instance deleted	Error (Terminating Setup) or CS instance deleted note 4	Error (1-Party)	1-Party	CS instance deleted or Error (Forward) note 1	Stable 1-Party /Stable_Multi_Passive_Party note 2	Stable 2-Party /Stable_ Multi_Party note 2

NOTE 1: Delete CS if passive leg status is joined. If passive leg 'pending' then Error case.
NOTE 2: State depends on the number of remaining legs.
NOTE 3: The O_BCSM is suspended at any valid DP for this CSCV state, else an error case.
NOTE 4: The T-BCSM is suspended at any valid DP for this CSCV state, else error case.

Table 32: Transition for Target CS

CSCV State (original state): ⇒ Operation: ⇓	Originating Setup	Originating 1-Party Setup	Stable 1-Party	Terminating Setup	1-Party	Stable 2-Party	Forward	Stable_Multi_Passive_Party	Stable Multi-Party
MoveLeg (c)	Error (Originating Setup)	Error (Originating 1-Party Setup)	Stable 2-Party	Error (Terminating Setup)	Error (1-Party)	Error (Stable 2-Party)	Error (Forward)	Stable Multi-Party	Error (Stable Multi-Party)
MoveLeg (p)	Stable 2-Party	Error (Originating 1-Party Setup)	Stable_Multi_Passive_Party	Error (Terminating Setup)	Stable 2-Party	Stable Multi-Party	Error (Forward)	Stable_Multi_Passive_Party	Stable Multi-Party

NOTE: State depends on the number of remaining legs.

11.31.3.2   Error handling

NOTE: When a CPH operation is received in an inappropriate CSCV state, a TaskRefused error shall be returned.

Error handling for the operation related errors is described in clause 13 and the TCAP services used for reporting operating errors are described in clause 15.

11.32   ReleaseCall procedure

11.32.1   General description

This operation is used by the SCF to tear down an existing call segment or all segments at any phase of the call for all parties involved in the relevant call segment(s).
This operation may not be sent to an assisting SSF, except in the case of hand-off procedure.
If a 'timed disconnect' is requested, an application timer 'Tdisconnect' is started on CSA level in SSF upon receipt of this operation. The execution of the ReleaseCall is postponed until the timer 'Tdisconnect' expires. When the timer Tdisconnect expires the ReleaseCall is handled as "immediate request" as described in "**FSM for CSA**" in clause 8.

11.32.1.1   Parameters

11.32.1.1.1   Argument Parameters

The operation argument consists of the following alternatives. These parameters are defined in clause 12.

- initialCallSegment;
  This alternative indicates that the initial CS shall be released. It comprises the following sub-parameters:

    - cause; or

- callSegmentToRelease;
  This alternative indicates that a single CS shall be released. It comprises the following sub-parameters:

    - callSegment;

    - releaseCause;

    - forcedRelease; or

- allCallSegments;
  This alternative indicates that all CSs shall be released. It comprises the following parameters:

    - releaseCause;

    - timeToRelease;

- forcedRelease.

11.32.2   Invoking entity (SCF)

11.32.2.1   Normal procedure

SCF Preconditions:

  (1) A relationship exists between the SCF and the SSF.

  (2) **FSM for CS** is in the state "Preparing SSF instructions" or the state "Waiting for Notification or Request".

  (3) If an SLP wants to ensure that a call will be released immediately on receipt of the ReleaseCall operation, regardless of the **FSM for CS** state in the SSF, the SLPI shall indicate that forced release is requested, i.e. parameter forcedRelease is to be included with the value set to "TRUE ". The forced release applies for the requested call segment(s) within the Call Segment Association.

SCF Postcondition:

(1) **FSM for CS** moves to the state "Idle", if neither "CallInformationReport" nor "ApplyChargingReport" has to be received from the SSF. All resources (e.g. queue) related to the call are released by the SCF; or
the state "Waiting for Notification or Request", if a "CallInformationReport" or "ApplyChargingReport" still has to be received from the SSF.

11.32.2.2   Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.32.3 Responding entity (SSF)

11.32.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) For CAMEL the gsmSSF is in the state" Waiting for Instructions" or state "Monitoring".

(3) If timeToRelease parameter is included a timer is started on CSA level in initiating SSF, i.e. the time the SSF waits before releasing the call. The timer is set upon receipt of the ReleaseCall operation.
When the timer expires, the ReleaseCall operation is distributed to all CSs in the CSA. Hereby the forcedRelease parameter is evaluated, if forced release is requested the call release is in any case to be performed immediately, otherwise the default release procedure applies.

(4) The **FSM for CS/Handed-off SSF FSM** is in any state, except Idle. If forced release is indicated, the call release is executed immediately, otherwise the default release procedure applies.
For the **FSM for CS** states in which the operation is accepted see table 21 and for the **Handed-off SSF FSM** see clause 8.4.

SSF Postcondition:

(1) The **FSM for CS** in initiating SSF moves to "Idle" state, after sending any outstanding reports "CallInformationReport" or "ApplyChargingReport".
Possible armed EDPs are ignored for the released CS(s). All connections and resources related to the CS or CSA are released. See table 21 "FSM for CS transition table" for the actions performed; or
The **Hand-off FSM** in assisting SSF moves to "Idle" state, after sending any outstanding report "ApplyChargingReport". All related connections and resources are released.
See clause 8.4 for the **Handed-off SSF FSM** for actions performed.

The default release procedure implies:

If the **FSM for CS**/**Assist Hand-off FSM** is in one of the states "Waiting for Instructions" or "Monitoring" the ReleaseCall operation is executed immediately.
If the **FSM for CS** in initiating SSF is in a state where user interaction is ongoing, i.e. in one of the following states:
"Waiting for End of User Interaction(WFI)"; or
"Waiting for End of User Interaction (MON)"; or
"Waiting for End of Temporary Connection(WFI)"; or
"Waiting for End of Temporary Connection(MON)"; or
the **Hand-off FSM** in assisting SSF is in state "Waiting for End of User Interaction"
then the user interaction is to be ended before the ReleaseCall is executed, i.e. the ReleaseCall operation is buffered until leaving the user interaction state for the concerned call segment.

11.32.3.2　　Error Handling

Operation related error handling is not applicable, due to class 4 operation.

11.33　ReportUTSI procedure

11.33.1　General description

This operation is used to notify the SCF of an USI previously requested by the SCF in a RequestReportUTSI operation.

This description covers the application of this operation for a call related transaction.

NOTE:　Refer to EN 301 931-1 for a definition of the term "User" in the context of the OCCRUI mechanism. A User to Service Information (USI) refers to either a UTSI or a STUI.

11.33.1.1　　Parameters

11.33.1.1.1　　　Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- uSIServiceIndicator.

- legID.

- uSIInformation.

- extensions.

NOTE:　For the applied leg numbering for legID refer to RequestReportBCSMEvent for a call associated transaction.

11.33.2　Invoking entity (SSF)

11.33.2.1　　Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The **FSM for CS** is in any state except "Idle".

(3)　The **SSF_USI FSM** is in the state "Monitoring USI".

SSF Postconditions:

(1) The **FSM for CS** remains in the same state.

(2) The **SSF_USI FSM** remains in the same state.

11.33.2.2　　Error handling

Operation related error handling is not applicable, due to Class 4 operation.

11.33.3 Responding entity (SCF)

11.33.3.1 Normal procedure

SCF Preconditions:

(1) A relationship exists between the SSF and the SCF.

(2) The **SCF_USI FSM** is in the state "Monitoring USI".

SCF Postconditions:

(1) SLPI execution continues.

(2) The **SCF_USI FSM** remains in the same state.

11.33.3.2 Error handling

Operation related error handling is not applicable, due to Class 4 operation.

11.34 RequestCurrentStatusReport procedure

11.34.1 General description

This operation is used to request the SSF to report the current status (busy or idle) of a particular termination resource.

11.34.1.1 Parameters

11.34.1.1.1 Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- resourceID.
  This parameter comprises the following alternatives:

  - lineID; or

  - facilityGroupID; or

  - facilityGroupMemberID; or

  - trunkGroupID.

11.34.1.1.2 Result Parameters

The operation result argument consists of the following parameters. These parameters are defined in clause 12.

- resourceStatus.

- resourceID.
  This parameter comprises the following alternatives:

  - lineID; or

  - facilityGroupID; or

  - facilityGroupMemberID; or

  - trunkGroupID;

- extensions.

11.34.2 Invoking entity (SCF)

11.34.2.1   Normal Procedure

SCF Preconditions:

   (1) The SLPI has determined that a "Request Current Status Report" operation has to be sent.

   (2) The **Status Report FSM** of the SCME is in the state "Status Report Idle".

SCF Postcondition:

   (1) The **Status Report FSM** of the **SCME** is in the state "Waiting for SSF Resource Status Report".

When the SLPI requires the current status of a physical termination resource, the SCF sends "Request Current Status Report" operation to the SSF to request the SSF to determine the status of the specific termination resource. Then the **SCME Status Report FSM** moves to the state "Waiting for SSF Resource Status Report" from the state "Status Report Idle". The **SCME Status Report FSM** is returned to the state "Status Report Idle", when the SCF receives a response from the SSF.

This operation is used outside the context of a call.

11.34.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.34.3 Responding entity (SSF)

11.34.3.1   Normal Procedure

SSF Precondition:

   (1) The **SSME FSM** is in state: "IdleManagement".

SSF Postconditions:

   (1) The **SSME FSM** is in the state "Idle Management".

   (2) After the SSF has determined the status of the specific termination resource the Result is sent to the SCF in RequestCurrentStatusReportResult.

On receipt of this operation the SSF determines the status of specific termination resource, and sends RequestCurrrentStatusReportResult as ReturnResult of this operation with the result of the status to the SCF. If an error occurs (e.g. the SSF can not find the specific termination resource), then the SSF sends ReturnError of this operation with the appropriate error type to the SCF.

11.34.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.35   RequestEveryStatusChangeReport procedure

11.35.1 General description

This operation is used to request the SSF start to monitor every change of the busy/idle status of a particular termination resource.

11.35.1.1   Parameters

11.35.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- resourceID.
  This parameter comprises the following alternatives:

    - lineID; or

    - facilityGroupID; or

    - facilityGroupMemberID; or

    - trunkGroupID;

- correlationID;

- monitorDuration;

- extensions.

11.35.1.1.2   Result Parameters

None.

11.35.2 Invoking entity (SCF)

11.35.2.1   Normal Procedure

SCF Preconditions:

(1) The SLPI has determined that a "Request Every Status Change Report" operation has to be sent.

(2) The **Status Report FSM** of the **SCME** is in the state "Status Report Idle".

SCF Postcondition:

(1) The **SCME Status Report FSM** is in the state "Waiting for SSF Resource Status Report".

When the SLPI requests to monitor every change of the busy/idle status of a physical termination resource, the SCF sends a "Request Every Status Change Report" operation to the SSF to monitor every change of the status of a particular termination resource. Then the **SCME Status Report FSM** is moved to the state "Waiting for SSF Resource Status Report" from the state "Status Report Idle". After this, in the case that the SCF receives a ReturnResult of this operation, the **SCME Status Report FSM** remains in same state and waits for a StatusReport operation. When the SCF receives a ReturnError of this operation, the **SCME Status Report FSM** is returned to the state "Status Report Idle".

This operation is used outside the context of a call.

11.35.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.35.3 Responding entity (SSF)

11.35.3.1 Normal Procedure

SSF Precondition:

(1) The **SSME FSM** is in the state "Idle Management".

SSF Postconditions:

(1) The **SSME FSM** is in the state "Non-Call Associated Treatment".

(2) A ReturnResult is sent to report the start monitoring of a particular termination resource.

On receipt of this operation, the SSF starts to monitor every change of the busy/idle status of a particular termination resource, and the SSF sends a ReturnResult of this operation to the SCF. If an error is occurred (e.g. the SSF can not find the specific termination resource), then the SSF sends a ReturnError of this operation with proper error type to the SCF.

The SSF continuously monitors every change of the busy/idle status of a particular termination resource until the timer which is specified by the monitorDuration parameter expires or the monitor is cancelled. When the SSF detects any change of status, or the cancellation/expiration of the requested monitor, the SSF sends the "Status Report" operation to the SCF.

11.35.3.2 Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.36 RequestFirstStatusMatchReport procedure

11.36.1 General description

This operation is used to request the SSF start to monitor the status of a particular termination resource for a change in the specific status (busy or idle).

11.36.1.1 Parameters

11.36.1.1.1 Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- resourceID;
  The parameter shall always be included.
  This parameter comprises the following alternatives:

  - lineID; or

  - facilityGroupID; or

  - facilityGroupMemberID; or

  - trunkGroupID;

- resourceStatus;
  The parameter shall always be included;

- correlationID;

- monitorDuration;

- extensions;

- bearerCapability.

11.36.1.1.2   Result Parameters

None.

11.36.2 Invoking entity (SCF)

11.36.2.1   Normal Procedure

SCF Preconditions:

(1) The SLPI has determined that a "Request First Status Match Report" operation has to be sent.

(2) The **Status Report FSM** of the **SCME** is in the state "Status Report Idle".

SCF Postcondition:

(1) The **Status Report FSM** of **SCME** is in the state "Waiting for SSF Resource Status Report".

When the SLPI requests to monitor the status of physical termination resource, the SCF sends "Request First Status Match Report" operation to the SSF to request the SSF to start to monitor a specific termination resource. Then the **SCME Status Report FSM** is moved to the state "Waiting for SSF Resource Status Report" from the state "Status Report Idle". After this, in the case that the SCF receives a ReturnResult of this operation, the **SCME Status Report FSM** remains in same state and waits for a StatusReport operation. When the SCF receives a ReturnError of this operation, the **SCME Status Report FSM** is returned to the state "Status Report Idle".

This operation is used outside the context of a call.

11.36.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.36.3 Responding entity (SSF)

11.36.3.1   Normal Procedure

SSF Precondition:

(1) The **SSME FSM** is in state "IdleManagement".

SSF Postconditions:

(1) The SSME FSM is in state "Non-Call Associated Treatment".

(2) A ReturnResult is sent to report the start monitoring of a particular termination resource.

On receipt of this operation, the SSF starts to monitor the status of a particular termination resource and the SSF sends a ReturnResult of this operation to the SCF. If an error has occurred (e.g. the SSF can not find a specific termination resource), then the SSF sends a ReturnError of this operation with proper error type to the SCF.

The SSF continuously monitors the status of a particular termination resource until the status matches a specific state, the timer which is specified by the monitorDuration parameter expires or the monitor is cancelled by the SCF. If the SSF detects the change of status in specific state or the monitor is cancelled or the monitor duration timer expires, the SSF sends the "Status Report" operation to the SCF.

11.36.3.2 Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.37 RequestNotificationChargingEvent procedure

11.37.1 General description

This operation is used to instruct the SSF how to manage the charging events which are received from other FEs not under the control of the service logic instance. The operation supports the options to cope with the interactions concerning the charging (refer to clause 16.6 "Interworking with other charge determination point" and scenarios A.2 and C.2). As several connection configurations may be established during a call a possibility exists for the RequestNotificationChargingEvent (RNC) operation to be invoked on multiple occasions. For each connection configuration a RNC may be used several times.

11.37.1.1 Parameters

11.37.1.1.1 Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- chargingEvents.
  This parameter indicates a list of chargingEvent out of a possible set (1 to n).
  For each chargingEvent it comprises the following sub-parameters:

  - eventTypeCharging;

  - monitorMode;

  - legID;

NOTE: This sub parameter indicates the leg id of the corresponding event type charging or event type tariff sub parameter.

  - eventTypeTariff;
    The event 'acknowledgement' corresponds to a tariff, add-on charge, start charge or stop charge send from SSF. It is not applicable for relayed charging messages.

11.37.2 Invoking entity (SCF)

11.37.2.1 Normal procedure

SCF Preconditions:

(1) A relationship exist between the SCF and the SSF.

(2) An SLPI has determined that a "RequestNotificationChargingEvent" has to be sent by the SCF.

SCF Postconditions:

(1) No **FSM** state transition.

(2) SLPI execution may continue.

The **SCSM FSM** for the CS is in state "Preparing CS Instruction" or is in state "Queuing FSM". This operation is invoked by the SCF if a SLPI results in the instruction of SSF how to cope with the interactions concerning the charging. This causes no **SCSM FSM** state transition.

11.37.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.37.3 Responding entity (SSF)

11.37.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exist between the SCF and the SSF.

(2) **FSM for CS**: State "Waiting for Instructions"; or
    FSM for CS:State "Waiting for End of User Interaction (WFI)"; or
    FSM for CS: State "Waiting for End of User Interaction (MON)"; or
    FSM for CS: State "Waiting for End of Temporary Connection (WFI)"; or
    FSM for CS:State "Waiting for End of Temporary Connection (MON)"; or
    FSM for CS: State "Monitoring"; or
    Assisting/Hand-off –FSM: State "Waiting for Instructions".

(3) The RNC operation is allowed for both controlling and passive legs. The indicated leg shall exist
    (leg status: pending or joined).

SSF Postcondition:

(1) No FSM state transition.

On receipt of this operation the SSF performs actions to cope with the interactions concerning the charging according to the information elements included in the operation. The requested charging event can be caused by: a) another SLPI; or b) another exchange. Irrespective of by what the charging event is caused the SSF performs one of the following actions on occurrence of the charging event according the corresponding monitorMode:

- Interrupted
  Notify the SCF of the charging event using "EventNotificationCharging" operation, do not process the event or propagate the signal. However call and existing charging processing will not be suspended in the SSF.

- NotifyAndContinue
  Notify the SCF of the charging event using "EventNotificationCharging", and continue processing the event or signal without waiting for SCF instructions (handled like EDP-N for BCSM events).

- Transparent
  Do not notify the SCF of the event. This ends the monitoring of a previously requested charging event.

Requested charging events are monitored until ended by a transparent monitor mode or until the leg is released or until the relationship is closed.

In the case that multiple "RequestNotificationChargingEvent" operations are received for the same connection configuration with the same "eventTypeCharging"/"eventTypeTariff" and "legID", only the last received "monitorMode" will apply.

11.37.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.38   RequestReportBCSMEvent procedure

11.38.1   General description

This operation is used to request the SSF to monitor for a call-related event (e.g. BCSM events such as busy or no answer), then send a notification back to the SCF when the event is detected or to disarm call-related event(s).

Event Detection Point (EDP) processing is always initiated by RequestReportBCSMEvent and the EDP may be acknowledged with an EventReportBCSM operation.

Every EDP must be explicitly armed by the SCF via a RequestReportBCSMEvent operation.
No implicit arming of EDPs at the SSF after reception of any operation (different from RequestReportBCSMEvent) from the SCF is allowed.

NOTE 1: If the RequestReportBCSMEvent requests arming of the current DP from which the call processing was suspended, the next occurrence of the DP encountered during BCSM processing will be detected (i.e. not the current one from which the call was suspended).

The detection point arming and disarming principle is as follows:

- All events for which filtering applies (abandon, midcall and disconnect) can be armed as well as for the controlling as passive legs depending on what direction (either from the party which is connected to the controlling or passive leg) events have to be captured. As an example, the Disconnect DP can be armed as well as for the controlling leg and the passive leg. In that case if a release request is received from the user it will be detected by the Disconnect DP armed for the controlling leg, while a release request from the remote parties shall be detected by arming the relevant passive leg Disconnect DP.

    - The O_Abandon DP can only be armed for the controlling leg in the O_BCSM and the T_Abandon can only be armed for the passive leg in the T_BCSM.

    - All DPs that can occur before DP-Analysed_Information in the O_BCSM are allowed for controlling leg only.

    - All DPs that can occur before PIC-Select_Facility in the T_BCSM are allowed for passive legs only.

    - The DP-Analysed_Information is allowed for controlling and passive legs.

    - All other DPs for which no filtering applies arming is allowed in the O_BCSM for passive legs only and in the T_BCSM for controlling leg only.

    - It is not possible to arm events on a surrogate leg in a CSCV.

    - When a leg in a CSCV becomes surrogate, all armed DPs associated with that leg automatically (implicit) become disarmed, e.g. at a CSCV transition from 'Terminating_Setup' to 'Forward'.

    - Only RRB arming requests for one new LegID ("leg to be created") is accepted and queued. The EDP arming request(s)may be provided in one or more RRB operations from the SCF (see note 3).

NOTE 2: The SLPI in the SCF becomes updated about the CSCV state evolution in the SSF via DP arming and DP event reporting as needed for a proper connection view in the SCF for the IN service logic execution.

NOTE 3: The following guidance applies to service logic in the SCF regarding the arming and queuing of BCSM events for a new LegID ("leg to be created") in the Connect operation:
A RRB for EDP arming in case of an unknown legID will be queued. In case where RRB (one or more) is sent on a not yet created leg, the service logic shall sent the operation for creation of the new leg in sequence following the RRB(s) and so indicate that only ONE queuing is needed per new LegID within the CSA. For example if two Connect operations have to be sent the correct sequence will be RRB(s)+Connect and RRB(s) + Connect. For details refer to "Queuing of BCSM events on receipt of an RRB operation in the CSA CV" as described within clause 6 under "Functional Procedures for Call Segment Association (CSA)".

NOTE 4: The above DP arming and disarming rules are applicable for one SLPI having a relationship with the SSF. For the DP event handling among multiple SLPIs refer to the clause on "DP Handling".

Table 33: DP ARMING and DISARMING Table for O-BCSM

O_BCSM	Controlling leg	Passive leg	Default_leg ID
Origination_Attempt DP **o1)**	-	-	-
Origination_Attempt_Denied DP	X	-	1
Orig._Attempt_Authorized DP	X	-	1
Collected_Information DP	X	-	1
Analysed_Information DP	X	X **o3)**	1
O_Term_Seized DP	-	X	2
Route_Select_Failure DP	-	X	2
Authorize_Route_Failure DP	-	X	2
O_Called_Party_Busy DP	-	X	2
O_No_Answer DP	-	X	2
O_Answer DP	-	X	2
O_Suspend DP	-	X	2
O_Re-Answer DP	-	X	2
O_Mid_Call DP	X	X	- **o2)**
O_Disconnect DP	X	X	- **o2)**
O_Abandon DP	X	-	1

o1)   Only applicable as TDP, because first DP that can be encountered cannot be armed as EDP.
o2)   The "legID" parameter shall be included.
o3)   Applicable for the passive leg created due to InitiateCallAttempt or Connect at Call Forwarding. For the InitiateCallAttempt operation a default LegID 1 applies for the **passive leg** to be created in order to secure IN CS-1 backward compatibility.

Nomenclature:
X = Arming applicable;
- = Not Applicable.

Table 34: DP ARMING AND DISARMING Table for T-BCSM

T_BCSM	Controlling leg	Passive leg	Default Leg ID
Termination_Attempt DP **t1)**	-	-	-
Termination_Attempt_Denied DP		X	1
Terminating_Attempt_Authorized DP	-	X	1
Facility_Selected_and_Available DP	X	-	2
Call_Accepted DP	X	-	2
T_Busy DP	X	-	2
T_No_Answer DP	X	-	2
T_Answer DP	X	-	2
T_Suspend DP	X	-	2
T_Re-Answer DP	X	-	2
T_Midcall DP	X	X	- **t2)**
T_Disconnect DP	X	X	- **t2)**
T_Abandon DP	-	X **t3)**	1

t1)   Only applicable as TDP, because first DP that can be encountered cannot be armed as EDP.
t2)   The "legID" parameter shall be included.
t3)   T_Abandon can only be armed for the passive leg.

11.38.1.1   Parameters

11.38.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- bcsmEvents;
  This parameter specifies a list of (1 to n) requested BCSM events. For each BCSM event, it comprises the following sub-parameters:

- eventTypeBCSM;

- monitorMode;

- legID;

- dPSpecificCriteria;
  It comprises the following alternatives:

  - numberOfDigits; or

  - numberOfDigitsTwo;
    It comprises the following sub-fields:

    - requestedNumberOfDigits;

    - minNumberOfDigits; or

- applicationTimer; or

- midCallControlInfo;
  This field specifies a list of (1 to n) requested specific midcall events. For each midcall event it comprises the following sub-fields:

  - midCallInfoType;
    It comprises the following sub-fields:

    - iNServiceControlCodeLow;

    - iNServiceControlCodeHigh;

    - midCallReportType;

- extensions.

When more than one event is requested the applied legID's in the list of requested bcsmEvent shall belong to the same CS. SCF will use the option "sendingSideID" only.

The following values for "legID" are assumed:

NOTE: The IN CS-1 definition of this parameter makes assumptions regarding the allocation of LegID values. With the introduction of Call Party Handling, these assumptions are no longer appropriate. For IN CS-3 the leg numbering is based on the following principles:

- legID = 1 is the controlling leg and legID = 2 is the passive leg in case the initial call segment created was an originating call segment (CS state 'Originating setup').
  For InitiateCallAttempt the default legID = 1 for the passive leg to be created.
  Additional legs can only be created by the SCF, in which case the SCF assigns the leg numbers.

- legID = 1 is the passive leg and legID = 2 is the controlling leg (i.e. inverse to the above) in case the initial call segment created was a terminating call segment (CS state 'Terminating setup'). Additional legs can only be created by the SCF, in which case the SCF assigns the leg numbers. For IN CS-1 implementations in the case of a mid call trigger it was assumed that legID = 2 was assigned to the party not causing the trigger and legID = 1 was assigned to the party causing the trigger.

The "legID" parameter shall always be included for the events O-MidCall, O-Disconnect, T-MidCall and T-Disconnect.

11.38.2 Invoking entity (SCF)

11.38.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The SLPI has decided that a request for an event report BCSM is needed.

(3) The **SCSM FSM** is in the appropriate state to send "RequestReportBCSMEvent".

SCF Postconditions:

(1) The **SCSM FSM** remains in the same state.

(2) SLPI execution continues.

(3) If all EDPs have been disarmed and neither a CallInformationReport nor an ApplyChargingReport is pending, the relationship with the concerned SSF is ended. If no other relationship persists, the FSM for CSA shall return to "Idle" state.

The *dPSpecificCriteria* sub-parameter indicates information specific of the event to be monitored.
It comprises different alternatives:

-    In case a number of digits is requested to be collected by the SSF, then all collected digits (including those digits collected previously at DP CollectedInfo) shall be sent to the SCF.
     The alternative *numberOfDigitsTwo* is used in case the SCF does not know the exact number of digits to collect, e.g. at unknown number length. If a complete number is determined by the CCF/SSF, the digit 'End Of Pulsing' (digit encoded 1111) shall be included (last digit) as part of the received CalledPartyNumber being reported as CollectedInfo by the SSF to the SCF. This is to inform the SLPI in the SCF that a complete number is provided. If complete number is determined in CCF/SSF and minimum number of digits has been collected, the received digits shall be reported to the SCF.
     If SCF receives less digits than requested in r*equestedNumberOfDigits*, it shall consider if complete number has been detected in CCF/SSF and reported and if so, shall not request more digits.

-    In case application timer for no answer event is specified, then this timer is expected to be shorter than the network no-answer timer.

NOTE: Service logic shall avoid clashes between user interaction event requests and iNServiceControlCode requested MidCall events.

EXAMPLE: Midcall control code 1234 monitored on same leg as an user interaction voice prompt for pin code 1234 shall be avoided.

11.38.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.38.3 Responding entity (SSF)

11.38.3.1   Normal procedure

SSF Precondition:

(1) The **FSM for CS** is in either the state "Waiting for Instructions" or in the state "Monitoring".
    If the **FSM for CS** is in state "Monitoring" only RequestReportBCSMEvent operations can be received with the monitorMode set to "transparent".

SSF Postconditions:

(1) The requested EDPs have been armed and/or disarmed as indicated.

(2) Previously requested events are monitored until ended by a transparent monitor mode, until the end of the call, until the EDPs are detected or implicitly disarmed or until the corresponding leg is released.

(3) The **FSM for CS** remains in the same state.

(4) If all EDPs have been disarmed and no "CallInformationReport" or "ApplyChargingReport" has been requested, the **FSM for CS** moves to the state "Idle".

11.38.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.39   RequestReportUTSI Procedure

11.39.1 General Description

This operation is used to request the SSF to monitor for the receipt of an USI information element with a given ServiceIndicator value, then send this USI back to the SCF when this information element is received.

This description covers the application of this operation for a call related transaction.

NOTE: Refer to EN 301 931-1 for a definition of the term "User" in the context of the OCCRUI mechanism. A User to Service Information (USI) refers to either UTSI or a STUI.

11.39.1.1   Parameters

11.39.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- requestedUTSIList;
  This parameter contains a list (1..n) of the requestedUTSI subparameter.
  This subparameter comprises the following fields:

  - uSIServiceIndicator;

  - uSImonitormode;

- extensions;

- legID.

NOTE: For the applied leg numbering for legID refer to RequestReportBCSMEvent for a call related transaction.

11.39.2 Invoking entity (SCF)

11.39.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SSF and the SCF.

(2) The SLPI has decided that it is necessary to monitor for the receipt of an USI information with a given ServiceIndicator value to precise.

(3) The **SCF_USI FSM** is in any state.

SCF Postconditions:

(1) SLPI execution continues.

(2) The **SCF_USI FSM** moves to the state "Monitoring USI" (if the USIMonitorMode is "monitoringActive") or to the state "Idle" (if the USIMonitorMode is "monitoringInactive").

11.39.2.2 Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.39.3 Responding entity (SSF)

11.39.3.1 Normal procedure

SSF Preconditions:

(1) A relationship exists between the SSF and the SCF.

(2) The **FSM for CS** is any state except "Idle".

(3) The **SSF_USI FSM** is in any state.

SSF Postconditions:

(1) The **FSM for CS** remains in the same state.

(2) The **SSF_USI FSM** moves to the state "Monitoring USI" (if the USIMonitorMode is "monitoringActive") or to the state "Idle" (if the USIMonitorMode is "monitoringInactive").

A *serviceIndicator* parameter value received from the SCF indicates the *serviceIndicator* value to be monitored for the indicated leg as the criteria for reporting USI information *(ReportUTSI operation)* to the SCF as received from a user. *The legID* parameter indicates the direction toward the user.

However, the SCF and the SRF may embed e.g. the "User Interaction" operations by means of USI Information, when the SCF uses the *ConnectToResource* operation to connect to an External SRF (Relay case).
A network specific value of the *serviceIndicator* parameter is allocated for the "External SRF connection": *SRF_Connection*. Once receiving the *RequestReportUTSI* operation from the SCF with a *serviceIndicator* parameter value set to *SRF_connection*, the SSF checks only this parameter to decide that this operation is related to the "SCF-External SRF communication". The SSF uses then the *legId* parameter to address the right external SRF (e.g. in case within the CSA several User Interaction are running in parallel on legs in different CSs). The same processing for *serviceIndicator* applies for the *ReportUTSI* and *SendSTUI* operation in the "SRF - SCF" direction.
See also EN 301 931-3 for further information.

11.39.3.2 Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.40 ResetTimer procedure

11.40.1 General description

This class 2 operation is used by the SCF to refresh the T_{SSF} application timer, in order to avoid the T_{SSF} time-out at the SSF.

11.40.1.1 Parameters

11.40.1.1.1 Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- timerID;
- timerValue;
- extensions;
- callSegmentID.

NOTE: When CS ID is not specified, the timer associated with the initial CS is assumed.

11.40.2 Invoking entity (SCF)

11.40.2.1 Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) An SLPI has determined by the $T_{SCF\text{-}SSF}$ guard timer expiration, that the "ResetTimer" operation has to be sent in order to avoid T_{SSF} time-out at the SSF.

SCF Postcondition:

(1) The SLPI resets the $T_{SCF\text{-}SSF}$ guard timer.

11.40.2.2 Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.40.3 Responding entity (SSF)

11.40.3.1 Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) Basic call processing has been suspended at a DP or user interaction (not CAMEL) is ongoing.

(3) The **FSM for CS** in initiating SSF is in the "Waiting for Instructions" state; or
in the "Waiting for End of User Interaction (WFI)" state; or
in the "Waiting for End of Temporary Connection (WFI)" state; or
in the "Waiting for End of User Interaction (MON)" state; or
in the "Waiting for End of Temporary Connection (MON)" state; or

The **Assing/Hand-off FSM** for the assisting SSF is in the "Waiting for Instructions" state; or
in the "Waiting for End of User Interaction" state.

NOTE: Whether the T_{SSF} is used or not in the user interaction states "Waiting for End of User Interaction" or in the state "Waiting for End of Temporary Connection" is network operator dependent.

SSF Postconditions:

> (1) The T_{SSF} timer has been reset.

> (2) The **FSM for the CS** (initiating SSF); or
> the **Assisting/Hand-off FSM** (assisting SSF)
> remains in the same state.

11.40.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.41   SelectFacility Procedure

11.41.1 General description

This operation is sent by the SCF to the SSF and requests the SSF to perform the terminating basic call processing actions to select the terminating line if it is idle.

11.41.1.1   Parameters

11.41.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- alertingPattern;

- extensions;

- serviceInteractionIndicatorsTwo;

- legID.
  LegID assigned to called party. When not provided, a default LegID of 2 is assumed.

11.41.2 Invoking Entity (SCF)

11.41.2.1   Normal Procedure

SCF Preconditions:

> (1) A relationship exists between a SCF and a SSF.

> (2) A SLPI has determined that a "SelectFacility" is to be sent by the SCF.

SCF Postcondition:

> (1) SLPI execution may continue.

In the **SCSM FSM** state "Preparing SSF Instructions", this operation is invoked by an SCF if the service logic results in the request to a SSF to route a call to a specific destination and to continue call processing at the Select_Facility_and_Present_Call PIC. If no event monitoring has been requested and no reports (CallInformationReport and ApplyChargingReport) have been requested in a previously sent operation, a **SCSM FSM** transition to the state "Idle" occurs. Otherwise, if event monitoring has been requested or any report (CallInformationReport and ApplyChargingReport) has been requested, the **SCSM FSM** transitions to the state "Waiting for Notification or Request".

11.41.2.2   Error Handling

If reject or error messages are received, then the **SCSM** informs the SLPI of the message and remains in the state "Preparing SSF Instructions".

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.41.3 Responding entity (SSF)

11.41.3.1   Normal Procedure

SSF Preconditions:

(1) A relationship exists between a SCF and a SSF.

(2) Call termination attempt has been initiated.

(3) Basic call processing has been suspended at a DP.

(4) The FSM for CS is in state "Waiting for Instructions".

SSF Postconditions:

(1) The SSF performs call processing action to select the terminating line.

(2) In the T-BCSM call processing shall resume at the Select_Facility PIC.

(3) If no EDPs have been armed and neither a CallInformationReport nor an ApplyChargingReport has been requested, the FSM goes to state "Idle" . Otherwise, the **FSM for CS** goes to state "Monitoring".

11.41.3.2   Error Handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.42   SendChargingInformation procedure

11.42.1 General description

This operation is used to instruct the SSF on the charging information to be sent by the SSF.
For CAMEL this operation is also used in correlation to the control of call duration.

Generally three different functions can be distinguished:

- Charging of the network access (e.g. 'calling line').
  The scenario supported is: A.2 (refer to clause 16 charging scenarios).

- Control online charge information provision to user access.

  The scenario supported is: C.2 (refer to clause 16 charging scenarios).

- Acknowledge charge messages received from other charge determination points (refer to clause 16.6.1 "Complete SCP control").

The sending of charging information can either be by charge pulses or signalling or internal e.g. if SSF is located in the Local Exchange (LE).

A possibility exists for the SendChargingInformation (SCI) operation to be invoked on multiple occasions.

NOTE: This operation has many PSTN/IN interactions.

11.42.1.1   Parameters

11.42.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- sCIBillingChargingCharacteristics;
  Interworking with signalling is based on ITU-T Recommendation Q.763;

- partyToCharge;

- extensions;

- tariffMessage;

- chargingControl;
  This parameter comprises the following sub-parameters tariffInfoFromSuccExchangeSCI and sSFdetermination:

    - tariffInfoFromSuccExchangeSCI;
      This subparameter comprises the following fields:

        - relayChargesFromDestination;

        - relayAOCFromDestination;

    - sSFdetermination;

- sCIBillingChargingCharacteristics for CAMEL:
  A detailed list is given by the CAMEL-SCIBillingChargingCharacteristics subparameter. The information is to
  be provided to the indicated call party. It is not send in the network direction.

11.42.2 Invoking entity (SCF)

11.42.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exist between the SCF and the SSF.

(2) An SLPI has determined that a "SendChargingInformation" has to be sent by the SCF.

SCF Postconditions:

(1) No FSM state transition.

(2) SLPI execution may continue.

The SCSM FSM is in state "Preparing SSF Instruction" or is in state "Queuing FSM". The SendChargingInformation
procedure shall be invoked by the SCF in accordance with the demands of the SLPI for relevant charging information.
If appropriate this information shall be sent back down the call path.

This causes no SCSM FSM state transition.

Generally three different scenarios can be distinguished from the initiating service logic point of view:

1) charging of the service user;

2) advice of charge;

3) request the SSP to send an acknowledgement via ISUP on an ISUP charge message previously received from
    backward direction.

In the following all three scenarios will be explained somewhat more in detail:

ad 1.:

The SCF either may:

- determine the charges itself; or

- relays charge messages from B-side; or

- require that the charges are to be determined on the base of the given destination.

In the first case the SCF acts as charge determination point as described in ES 201 296. In case of multiple connection configurations, the SCF may act as 'connection control point', that means the SCF translates the receipt of Answer- respectively Release messages from B-side to a SCI operation containing the StartCharging- or StopCharging parameter.

In the second case the SCF relays charge messages received from B-side. Therefore the corresponding charging messages must be requested by sending an RNC operation with monitorMode 'interrupt' previously. The SCF may also modify the information. In case of multiple connection configurations, the SCF acts in this case as 'connection control point' and translates the receipt of Answer- respectively Release messages from B-side to a SCI operation containing the StartCharging- or StopCharging parameter.

In the third case again two cases may be distinguished:

- The SSP (CCF) has to determine the charges and the handling for charge messages from B-side based on its own database. This shall either be done based on the destination routing address or based on the called IN number. SSP determination based on the destination routing address is only applicable for connections to a B-subscriber.

- Charge messages from B-side have to be forwarded to the partyToCharge unconditionally.

In both cases the SSF acts as a connection control point.

Note, that there are the following restrictions with respect to the forwarding of charge messages received from B-side:

- In case of forwarding charge messages from more than one outgoing leg to a special A-party the rules at the charge generation or registration point of the underlying basic network regarding multiple tariff determination have to be regarded. Dependent on the 'network identification' of the sender of the charge messages they could overwrite each other. According to the ES 201 296 no multiple tariff determination for one network operator is foreseen.

- It is only possible to forward charge messages from a special destination to the partyToCharge, if they are connected within the same Call Segment at the connection set-up.

ad 2.:

The SCF either may:

- determine the charges for the online charge information provision to user access itself; or

- relay corresponding charge messages from B-side; or

- require that the corresponding charges messages from B-side are to be forwarded at the SSP.

In the first case the SCF delivers tariff data for the online charge information provision to user access within the SCI operation. In this case the parameter 'tariff message' indicates 'charging tariff information' or 'add-on charging information' with sub-parameter 'charging control indicator' set to value 'advice of charge'.

In the second case the SCF relays corresponding charge messages received from B-side. Therefore the same mechanism as described above for 'ad 1.' apply.

The third case, that means the SCF requires that the charges are to be forwarded from B-side, is only applicable for connections to a B-subscriber. Note, that there are the same restrictions with respect to the forwarding of charge messages that receive from B-side as described above for case 'ad 1.'.

ad 3.:

This scenario is applicable in case, the SCF has received an ENC operation sent by the SSP due to a received ISUP charge message (monitoring of charging events has to be required by the service logic before, refer to the detailed RNC/ENC procedure descriptions). After the charging information received by the ENC operation has been evaluated by the service logic, the SCF shall send a SCI operation containing the parameter 'charge acknowledge information' to request the SSF to acknowledge the received ISUP charge message.

11.42.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.42.3 Responding entity (SSF)

11.42.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) **FSM for CS** State "Waiting for Instructions"; or
FSM for CS State "Waiting for End of User Interaction(WFI)"; or
FSM for CS State "Waiting for End of User Interaction(MON)"; or
FSM for CS State "Waiting for End of Temporary Connection (WFI)"; or
FSM for CS State "Waiting for End of Temporary Connection (MON)"; or
FSM for CS State "Monitoring"; or

Assisting/hand-off FSM in the State: "Waiting for Instructions".

SSF Postcondition:

(1) No FSM for CS or Assisting/hand-off FSM state transition.

On receipt of this operation the SSF performs actions to send the charging information. The sending of charging information can either be by charge pulses or signalling or internal e.g. if SSF is located in LE. This operation has much PSTN/IN interactions.

(A) Processing of parameter 'tariff message'

Forwarding of parameter 'tariff message' via ISUP is described within ES 201 296.

There are the following peculiarities that result from the special function split between SSF and SCF. The SCI is always mapped onto the ISUP-APM (Application Transport Message). The Application Context Identifier subfield shall be set to, 'charging ASE'. The SCI-parameter 'tariff message' is mapped to the Encapsulated Application Information subfield.

- Handling of Application Transport Instruction Indicators (ATII).

  This cannot be transmitted over INAP, so the SSP shall set-up the values to be transmitted in ISUP. The provision of values 'release call' or 'continue call' is possible and which value is used is a network provider option.

  The values can depend on the kind of TariffMessage received in the SCI.

- Handling of Acknowledgement Timer (Tcrga).

  In case, the parameter 'tariff message' includes a tariff-, add-on-, start- or stop charge sub-parameter the SSF has to take over following signalling specific tasks of the charge determination point: Start the timer Tcrga awaiting the confirmation, respectively, representing the acknowledgement. On receipt of the acknowledgement the timer Tcrga shall be cancelled.

NOTE 1: If Tcrga is still running, awaiting the acknowledgement to a previous SCI containing tariff- or add-on charge, a new request cannot be issued until an acknowledgement has been received or Tcrga expires.

- The exceptional procedures:

  - on expiry of timer Tcrga; or

  - on receipt of a CRGT, AOCRG, START or STOP confirmation primitive with the indication "not accepted"; or

  - on receipt of the Charging_Error primitive;

  - have to be performed at the SSF (either releases the call or continues the call, this is a network provider option) except of the case, that the SCF has requested the SSF to have itself the whole control (that means, via RNC the 'event type tariff' 'charging acknowledgement information' or 'charging acknowledge timer expired' with monitor mode 'interrupted' has been received).

- Handling of Charging Reference Identifiers (CRI).

In case the parameter 'tariff message' includes a charging acknowledgement information both charging reference identifiers are provided from SCF.

Otherwise, the parameter 'tariff message' includes a tariff-, add-on-, start- or stop charge, the SSF has always to supply the 'destination identification' according the rules described within ES 201 296 for the charge determination instance.

(B) Charge determination at the SSP

The corresponding SCI operation contains the sub-parameter sSFdetermination with value destinationRoutingAddress or calledInNumber.

In this case the tariff shall be determined at the SSP based on either the destination routing address digits received in the corresponding related connection-setup operation Connect or the destination routing address digits which are used for connection-setup in context with one of the operations Continue and ContinueWithArgument or based on the called IN number which is the calledPartyNumber received in the InitialDP operation. In this scenario the SSP acts as 'charge determination point' itself exactly as described within ES 201 296.

NOTE 2: In case, that the signalling capabilities of the underlying network does not allow the ISUP signalling according to ES 201 296, the interworking has to be defined in a network specific way.

It is also possible that the SSP detects, during trying to determine the charges, that the tariff shall be determined in a succeeding exchange.

(C) Relay of charge messages

Charge messages received from another destination (e.g. from a higher exchange, an other SSF-SCF instance or an international gateway) shall be forwarded to the origin indicated within 'partyToCharge'.

This is only supported, if the SCF has not requested the SSF to have the control (via RNC operation with 'event type tariff' chargingTariffInformation, addOnchargingInformation, startCharging or stopCharging and monitor mode 'interrupted'). In case of interworking with the FurnishChargingInformation regarding the tariffInfoFromSuccExchange parameter refer to clause 16, table 44.

If the SCI indicates with parameter 'relayChargesFromDestination' that charge messages received from a succeeding exchange shall be forwarded or the SSP itself has determined that charges are to be determined in a succeeding exchange (see (B)), all charge messages are to be forwarded.

Table 35 depicts the possible SCI control parameters and the interworking with SSP based charge determination and the resulting handling of charge messages from succeeding exchange.

Table 35: SCI control parameters and the interworking with SSP

Values of sub-parameter relayChargesFromDestination within parameter tariffInfoFromSuccExchangeSCI	Values of parameter sSFdetermination	Handling of charge messages received from B-side which are applicable for 'subscriber charge'
noIndication	noDetermination	no relay
	destinationRoutingAddress or calledInNumber	relay as determined at SSP
relay	noDetermination	relay
	destinationRoutingAddress or calledInNumber	relay
noRelay (note)	noDetermination (note)	no relay
	destinationRoutingAddress or calledInNumber	no relay
NOTE: This are the default values as defined via ASN.1 if the parameters are not supplied or even the parameter chargingControl is not present.		

APM messages containing CRGT or AOCRG are forwarded if the subscriberCharge parameter indicates 'subscriber-charge'.

If the SCI indicates with parameter 'relayAOCFromDestination' that charge messages related to AOC shall be forwarded, all APM messages containing CRGT or AOCRG are forwarded if the subscriberCharge parameter indicates 'advice-of-charge'.

The corresponding SCI operation has to relate to a connection-setup operation (CON, CUE or CWA).

The SSF acts as a connection control point in this case (see ES 201 296).

11.42.3.2   Error handling

For CAMEL TaskRefused:
In addition to the generic error handling noted below, this error shall be indicated when:

-    a tariffSwitchInterval is indicated when a previously received tariffSwitchInterval is pending.

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.43   SendSTUI procedure

11.43.1 General description

This operation is used to request the SSF to forward an STUI with a given ServiceIndicator value to the User (indicated by leg ID).

This description covers the application of this operation for a call related transaction.

   NOTE: Refer to EN 301 931-1 for a definition of the term "User" in the context of the OCCRUI mechanism and to EN 301 931-3 for the application with an SRF connection.

11.43.1.1   Parameters

11.43.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- uSIServiceIndicator;

- legID;

- uSIInformation;

- extensions.

NOTE: For the applied leg numbering for legID refer to RequestReport BCSMEvent for a call related transaction.

11.43.2 Invoking entity (SCF)

11.43.2.1   Normal procedure

SCF Preconditions:

(1) A relationship exists between the SSF and the SCF.

(2) The SLPI has decided that it is necessary to send to the User a USI Information.

(3) The **SCF_USI FSM** is in any state.

SCF Postconditions:

(1) SLPI execution continues.

(2) The **SCF_USI FSM** remains in the same state.

11.43.2.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.43.3 Responding entity (SSF)

11.43.3.1   Normal procedure

SSF Preconditions:

(1) A relationship exists between the SSF and the SCF.

(2) The **FSM for CS** is any state except "Idle".

(3) The **SSF_USI FSM** is in any state.

SSF Postconditions:

(1) The **FSM for CS** remains in the same state.

(2) The **SSF_USI FSM** remains in the same state.

On receipt of this operation, the SSF will forward the STUI Information element to the User (identified by the legID).

11.43.3.2   Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.44   ServiceFilteringResponse procedure

11.44.1   General description

This operation is used to report the values of counters specified in a previous sent "ActivateServiceFiltering" operation to the SCF.

11.44.1.1   Parameters

11.44.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- countersValue;

- filteringCriteria;
  this parameter comprises the following alternatives:

  - serviceKey; or

  - addressAndService;

- extensions;

- responseCondition.

11.44.2   Invoking entity (SSF)

11.44.2.1   Normal procedure

SSF Preconditions:

(1) Service filtering is running and the interval time is expired and a call is received; or

(2) Service filtering is running and the threshold value is reached; or

(3) Service filtering has been finished (duration time expired or stop time met); or

(4) The operation "ActivateServiceFiltering" is received and encounters an active service filtering entity.

SSF Postcondition:

(1) Service filtering proceeds or is ended depending on the duration time.

The SSF sends the "ServiceFilteringResponse" operation to the SCF. The "filteringCriteria" parameter is provided to enable the addressing of the concerned service logic at the SCF.

Before "ServiceFilteringResponse" is sent due to SSF precondition 1 or 2, it is checked whether call gapping criteria are met for the related call (e.g. InitialDP) If so, the "ServiceFilteringResponse" is not sent and the counting continues without resetting the counters. The last "ServiceFilteringResponse" (stop time is met or duration time expired) is sent without checking any call gap criteria.

After sending "ServiceFilteringResponse" the service filtering counters are reset.

If service filtering proceeds after sending "ServiceFilteringResponse" (e.g. interval time expired) the SSME-FSM remains in the state "Non-Call Associated Treatment".

If service filtering is stopped after sending "ServiceFilteringResponse" (duration time expired or stop time is met) then the **SSME-FSM** moves to the "Idle Management" state. All allocated resources are released, i.e. the **SSME-FSM** is removed as well.

11.44.2.2   Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.44.3   Responding entity (SCF)

11.44.3.1   Normal procedure

SCF Preconditions:

    (1) Service filtering is running.

    (2) The **Service Filtering FSM** of the **SCME** is in the state "Waiting for Service Filtering Response".

SCF Postcondition:

    (1) The **Service Filtering FSM** of the **SCME** forwards the received counter values to the SLPI.

The operation is handled by the **Service Filtering FSM** part of the **SCF Management Entity (SCME)**. The **SCME** passes the received counter values to the SLPI where they are added to previously received counter values.

The "filteringCriteria" parameter as provided in "ServiceFilteringResponse" is used to address the **SCME** and the concerned service logic instance.

The **Service Filtering FSM** of the **SCME** remains in the state "Waiting For SSF Service Filtering Response" until the internal service filtering duration time in the SLPI expires. Then the SLPI informs the **SCME** about timer expiration. Now the **Service Filtering FSM** part of the **SCME** moves to the state "Service Filtering Idle".

11.44.3.2   Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.45   SetServiceProfile

11.45.1 General description

This operation is used in the context of a call to activate/de-activate a service profile for the party associated with the controlling leg. A profile list including an associated action indicator is provided and applicable to the BCSM for which triggering occurred for the controlling party as long as the associated BCSM exists in the call.

11.45.1.1   Parameters

11.45.1.1.1   Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

    - inProfiles;
       this parameter specifies a list (1 to n) of INProfile. For each INProfile it comprises the following sub-parameters:

        - tDPIdentifier;

        - dPName;

        - actionOnProfile;

        - extensions;

- extensions
  (on argument level).

11.45.2 Invoking Entity (SCF)

11.45.2.1 Normal Procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) SLPI detects that triggers have to be activated.

SCF Postconditions:

(1) The **SCSM FSM** remains in the same state.

(2) SLPI execution continues.

11.45.2.2 Error Handling

Generic error handling for the operation related error are described in clause 13 and the TC services used for reporting operation errors are described in clause 15.

11.45.3 Responding Entity (SSF)

11.45.3.1 Normal Procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The **FSM for CS** is in any state, except "idle".

SSF Postconditions:

(1) The corresponding triggers are activated/de-activated for the call.

(2) The **FSM for CS** remains in the same state.

11.45.3.2 Error Handling

Generic error handling for the operation related error are described in clause 13 and the TC services used for reporting operation errors are described in clause 15.

11.46 SplitLeg procedure

11.46.1 General description

This operation is used to request the SSF to separate one party from its Call Segment and place it in a new associated CS. This operation is the inverse of the MergeCallSegments operation.

In splitting the specified leg, the conditions of the leg: the armed EDPs, the ApplyChargingReport pending, the EventNotificationCharging pending, and the CallInformationReport pending, are also applied for the same leg after split.

11.46.1.1 Parameters

11.46.1.1.1 Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- legToBeSplit;

- newCallSegment;

- extensions.

11.46.1.1.2 Result Parameters

None.

11.46.2 Invoking entity (SCF)

11.46.2.1 Normal procedure

SCF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) An SLPI has determined that a call party shall be split from its current Connection Point.

SCF Postconditions:

(1) SLPI execution may continue.

(2) The **SCSM FSM** remains in the same state.

(3) The **FSM for CS** is in state "Preparing CS Instructions".

11.46.2.2 Error handling

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.46.3 Responding entity (SSF)

11.46.3.1 Normal procedure

SSF Preconditions:

(1) A relationship exists between the SCF and the SSF.

(2) The **CSA** CV is in state "Multi-Segment" or "One-Segment ".

(3) The **FSM for CS** is in the state "Waiting For Instructions" or "Monitoring".

(4) The leg to be split has the status "joined".

(5) The **CSCV** is in a valid connection view state according to the **CSCV** transition tables below.

SSF Postconditions:

(1) The SSF performs the necessary actions to separate the indicated leg from its original Call Segment and place it in a new associated Call Segment.

(2) The **FSM for CS** of the new Call Segment moves to the "Waiting For Instructions" state.

(3) The **FSM for CS** for the source Call Segment moves or remains in the "Waiting for instructions" state.

(4) The associated BCSM instances within the two involved Call Segments will move from the PIC to the DP of the corresponding O_/T_MidCall DP, when not already suspended at a DP. Note that no MidCall EDP will be reported for this case.

(5) A Return Result is sent immediately after the successful change of the leg configuration is executed.

NOTE: This allows the SCF to be updated with the established connection view and to cater for possible interference problems with signalling events.

(6) The **CSA** CV is in state "Multi-Segment".

Table 36: Transition for Source CS

CSCV State (original state): ⇒ Operation: ⇓	Originating Setup	Originating 1-Party Setup	Stable 1-Party	Terminating Setup	1-Party	Stable 2-Party	Forward	Stable_Multi_Passive _Party	Stable Multi-Party
SplitLeg (c)	Error (Originating Setup)	Error (Originating 1-Party Setup)	Error (Stable 1-Party)	Error (Terminating_Setup)	Error (1-Party)	Stable_1_Party	Error (Forward)	Error (Stable_Multi_ Passive_Party)	Stable_Multi_ Passive_Party
SplitLeg (p)	Error (Originating Setup)	Error (Originating 1-Party Setup)	Error (Stable 1-Party)	Error (Terminating_Setup)	Error (1-Party)	1-Party		Error (Forward)	Stable 1-Party/ Stable_Multi_ Passive_Party note 2	Stable 2-Party /Stable Multi-Party note 2
NOTE 1: Void.									
NOTE 2: The state depends on the number of remaining legs.									

Table 37: Transition for New CS

CSCV State (original state): ⇒ Operation: ⇓	Null
SplitLeg (c)	1-Party
SplitLeg (p)	Stable 1-Party (see note)
NOTE: The receipt of a SplitLeg (p) operation for the Stable 2-Party and Stable Multi-Party states shall create a new Call Segment in the CSCV Stable_1_Party state. The receipt of a SplitLeg (p) operation for the Stable_Multi_Passive_Party state shall create a new Call Segment in the CSCV Stable_1_Party state.	

11.46.3.2 Error handling

NOTE: When a CPH operation is received in an inappropriate CSCV state, a TaskRefused error shall be returned.

Generic error handling for the operation related errors are described in clause 13 and the TCAP services used for reporting operation errors are described in clause 15.

11.47 StatusReport procedure

11.47.1 General description

This operation is used to notify the result of monitoring that is requested by "RequestFirstStatusMatchReport" or "RequestEveryStatusChangeReport" operation to the SCF.

11.47.1.1 Parameters

11.47.1.1.1 Argument Parameters

The operation argument consists of the following parameters. These parameters are defined in clause 12.

- resourceStatus;

- correlationID;

- resourceID;
  this parameter comprises the following alternatives:

  - lineID; or

  - facilityGroupID; or

  - facilityGroupMemberID; or

  - trunkGroupID;

- extensions;

- reportCondition.

11.47.2 Invoking entity (SSF)

11.47.2.1 Normal Procedure

SSF Precondition:

(1) The **SSME FSM** is in the state "Non-Call Associated Treatment".

SSF Postcondition:

(1) The **SSME FSM** is in one of the following states:

- state: "IdleManagement"; or

- state: "Non-Call Associated Treatment".

The SSF sends the "StatusReport" operation to the SCF, when the following events have occurred:

- The SSF finds the change/match of status in specific state.

- The SSF receives the "CancelStatusReportRequest" operation from the SCF.

- The monitor duration timer expiration.

The SSF uses the "reportCondition" parameter for indicating the occurred event to the SCF. For further details on setting "reportCondition" value, refer to the reportCondition parameter definition.

After the SSF sends this operation, the **SSME FSM** should move to the state "Idle Management" unless a further StatusReport for RequestEveryStatusChangeReport is needed, in which case the **SSME FSM** should remain in the state "Non Call Associated Treatment".

This operation is used outside the context of a call.

11.47.2.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

11.47.3 Responding entity (SCF)

11.47.3.1 Normal Procedure

SCF Precondition:

(1) The **Status Report FSM** of the **SCME** is in the state "Waiting for SSF Resource Status Report".

SCF Postconditions:

(1) The **Status Report FSM** of the **SCME** moves to the state "Status Report Idle", if the StatusReport is one of the following:

- StatusReport for RequestFirstStatusMatchReport when the resource status matches; or

- StatusReport for reporting the monitor duration timer expiration; or

- StatusReport for reporting the cancellation of the monitor.

(2) The **SCME Status Report FSM** moves to the state "Waiting for SSF Resource Status Report" if the StatusReport is for reporting the resource status change in response to the RequestEveryStatusChangeReport.

(3) The SCF notifies the SLPI of the result of resource monitoring in the SSF.

The distinction among the status match/change, monitor cancellation, and the timer expiration is done by the "reportCondition" parameter in the StatusReport operation. For further details on setting "reportCondition" value, refer to the reportCondition parameter definition.

On receipt of this operation the SLPI which is expecting this operation will continue.

11.47.3.2   Error handling

Operation related error handling is not applicable, due to class 4 operation.

12   Parameter Descriptions

This clause defines the parameters used in the operations procedures as specified in clause 11.

12.1   AChBillingChargingCharacteristics

This parameter specifies the charging related information to be provided by the SSF and the conditions on which this information has to be reported back to the SCF via the "ApplyChargingReport" operation. Its content is network operator specific.

12.2   AChChargingAddress

This parameter indicates the charging address, leg, srfCallSegment or bNCF (basic Network Charging Function), to which the "ApplyCharging" operation shall be applied. If not present, then it is applied to the default legID 1
This parameter comprises the following alternatives:

- legID:
  this alternative indicates that the charging supervision is associated to the specified leg.
  The charging scenarios supported for this parameter is B1and B2 (refer to clause 16 "Charging Scenarios supported by Core INAP"); or

- srfCallSegment:
  this alternative indicates that the charging supervision is associated to the specified call segment. The parameter indicates that the charging is related to a User Interaction SRF connection for a call segment. The charging scenarios supported for this parameter isB1and B2 (refer to clause 16 "Charging Scenarios supported by Core INAP"); or

- bNCF:
  this alternative indicates that the charging supervision is associated to the Basic Network Charging Function (calling line charging). The charging scenarios supported for this parameter is A.1 and A.2 (refer to clause 16 "Charging Scenarios supported by Core INAP"). This parameter can be used for pulse and cost supervision of the BNCF if the SSF is in the same switch.

12.3   ActionIndicator

This parameter indicates the Trigger Detection Point (TDP) action to be performed, e.g. activate a TDP; deactivate a TDP or interrogate the current status of a TDP.

12.4   ActionPerformed

This parameter indicates the result of the Trigger Detection Point (TDP) management action performed e.g. TDP activated; deactivated or actual status interrogated (active/inactive).

12.5 AdditionalCallingPartyNumber

This parameter contains the calling party number provided by the access signalling system of the calling user (e.g. provided by a PBX). Refer to ITU-T Recommendation Q.763 for the actual definition of this parameter.

12.6 AlertingPattern

This parameter contains an indication for a specific pattern that is used to alert a subscriber (e.g. distinctive ringing tones, etc). It only applies if the network signalling support this parameter or if SSF is in the terminating local exchange for the subscriber. Refer to ITU-T Recommendation Q.931 Signal parameter or TS 129 002 for the actual definition of this parameter.

12.7 AllCallSegments

This parameter indicates all Call Segments within the Call Segment Association.

12.8 AllRequests

This parameter indicates that all active requests for EDP report; "ApplyChargingReport" and "CallInformationReport" should be cancelled for all call segments.

12.9 AllRequestsForCallSegment

For a definition of this parameter refer to the 'CallSegmentID' clause. It indicates that all active requests for EDP report; "ApplyChargingReport" and "CallInformationReport" should be cancelled for the specified call segment.

12.10 AssistingSSPIPRoutingAddress

This parameter indicates the destination address of the SRF for assist procedure.
The "assistingSSPIPRoutingAddress" contains a number for routing of the call to the SRF and may contain embedded within it, a "correlationID" and "scfID", but only if "correlationID" and "scfID" are not specified as separate parameters. For a definition of "correlationID" and "scfID" refer to the clauses 'CorrelationID' and 'ScfID' describing these parameters.

12.11 BackwardGVNS

This parameter contains information sent backward to the originating side about how the VPN call is terminated at the terminating side. Refer to ITU-T Recommendation Q.762 for the actual definition of this parameter.

12.12 BcsmEvents

This parameter specifies a list of (1 to n) requested BCSM events. For each BCSM event, it comprises the following sub-parameters:

- eventTypeBCSM
  For a definition of this sub-parameter, refer to the 'EventTypeBCSM' clause.

- monitorMode
  For a definition of this sub-parameter, refer to clause 'MonitorMode'.
  It is used as follows: When the monitorMode is "interrupted", the event shall be reported as a request, if the "monitorMode" is "notifyAndContinue" the event shall be reported as a notification. When the "monitorMode" is "transparent" the event shall not be reported.

- legID
  For a definition of this sub-parameter, refer to clause 'LegID'.

- dPSpecificCriteria
  This sub-parameter indicates information specific of the event to be monitored. It comprises the following alternatives:

  - numberOfDigits
    This alternative indicates the number of digits to be collected by the SSF for the CollectedInfo event. If the indicated number of digits is collected, SSF reports the event to the SCF; or

  - numberOfDigitsTwo
    This field indicates the number of digits to be collected by the SSF for the CollectedInfo event when the SCF does not know the exact number of digits to be collected.
    If the indicated requested number of digits is collected or a complete number is determined (e.g. by end of dialling number analysis or interdigit timeout) by the CCF/SSF, the SSF is to report the collected information to the SCF;
    it comprises the following sub-fields:

    - requestedNumberOfDigits
      This field indicates the number of digits requested to be collected by the SSF for the CollectedInfo event.

    - minNumberOfDigits
      This optional field is to indicate the minimum number of digits to be collected by the CCF/SSF for the CollectedInfo event, in case the SCP does not know the exact number of digits to collect, e.g. in an open Numbering Plan; or

  - applicationTimer
    This alternative indicates the application timer for the NoAnswer event. If the user does not answer the call within the allotted time, the SSF reports the event to the SCF. This timer is expected to be shorter than the network no-answer timer is; or

  - midCallControlInfo
    This field specifies a list of (1 to n) requested specific midcall events, which are to be monitored by the CCF/SSF.
    Several midcall events can be required in parallel and may contain one or more control codes to define the specific events. For each midcall event it comprises the following sub-fields:

    - midCallInfoType
      It comprises the following subfields:

      - iNServiceControlCodeLow:
        This sub-field contains a single control code or the lower bound of a control code interval. Value "0" for single control code is used for hook flash for analogue line.

      - iNServiceControlCodeHigh:
        This sub-field contains the upper bound of a control code interval.

    - midCallReportType
      This sub-field indicates how a detected midcall event is to be reported by the SSF to the SCF.
      When this sub-field is not provided default of report in Monitoring state is applied.
      It comprises the following alternatives:

      - *inMonitoringState*: report event when **FSM for CS** is in state Monitoring; or

      - *inAnyState:* report event when **FSM for CS** is in any state, except idle (immediate report).

12.13 BCSMFailure

This parameter indicates the occurrence of a BCSM failure. It comprises the following sub-parameters:

- legID
  For a definition of this sub-parameter, refer to the 'LegID' clause. It is used to identify the released leg.

- reason
  This sub-parameter provides network specific information about the kind of error/exception (e.g. external or internal error or exception), if present. Its content is network operator specific.

- cause
  For a definition of this sub-parameter, refer to the 'Cause' clause. It is used to indicate the cause of releasing this specific entity. The cause may be used by the SCF to decide about the further treatment of the call.

12.14 BearerCapability

This parameter indicates the type of the bearer capability connection or the transmission medium requirements to the user. It comprises the following alternative sub-parameters. It is a network option to select one to be used:

- bearerCap:
  this alternative sub-parameter contains the value of the DSS1 Bearer Capability parameter (ITU-T Recommendation Q.931) in case the SSF is at local exchange level or the value of the ISUP User Service Information parameter (ITU-T Recommendation Q.763) in case the SSF is at transit exchange level.

  If two values for bearer capability are available at the SSF or if User Service Information and User Service Information Prime are available at the SSF the "bearerCap" shall contain the value of the preferred bearer capability respectively the value of the User Service Information Prime parameter. Refer to ITU-T Recommendation Q.1601; or

- tmr:
  this alternative sub-parameter contains the tmr value. Refer to ITU-T Recommendation Q.762 for the actual definition of this parameter.
  If two values for transmission medium requirement are available at the SSF or if Transmission Medium Requirement and Transmission Medium Requirement Prime are available at the SSF the "bearerCap" shall contain the value of the preferred transmission medium requirement respectively the value of the Transmission Medium Requirement Prime parameter. Refer to ITU-T Recommendation Q.1601.
  BusyCause.

For a definition of this parameter, refer to the 'Cause' clause. It contains a cause value that specifies why the called party was busy.

12.15 CalledDirectoryNumber

This parameter contains the directory number of the called party for the call.
Refer to ITU Recommendation Q.762 for the actual definition of this parameter.

12.16 CalledPartyBCDNumber

This parameter contains the number used to identify the called party in the forward direction. It may also include service selection information, including * and # characters.

12.17 CalledPartyNumber

This parameter contains the number (e.g. called directory number) used to identify the called party.
Refer to ITU-T Recommendation Q.762 for the actual definition of this parameter.

12.18   CallingGeodeticLocation

The parameter indicates the geographic co-ordinate of a calling party.
Refer to ITU-T Recommendation Q.762 for the actual definition of this parameter.
The excessive amount of data possible within this parameter may require segmentation of the INAP operation to send to the SCF.
The amount of data possible to be conveyed within this parameter from the SSF to the SCF could be limited; for example it may be considered to only support a relevant subset of all the shape descriptions.

12.19   CallingPartyBusinessGroupID

This parameter identifies the business group associated with the calling party. A business group represents a logical grouping of service subscribers, who share a set of service properties. The SCF can use this parameter to select SLPs based on the group and for authorization purposes. The network operators can specify that this parameter should be used if their particular network has the information available.

12.20   CallingPartyNumber

This parameter is used to identify the calling party for the call.
Refer to ITU-T Recommendation Q.762 for the actual definition of this parameter.

12.21   CallingPartysCategory

This parameter contains information sent in the forward direction indicating the category of the calling party.
Refer to ITU-T Recommendation Q.762 for the actual definition of this parameter.

12.22   CallReference

The Call Reference value is unique within one network.
Refer to ITU-T Recommendation Q.762.

12.23   CallReferenceNumber

This parameter gives the call reference number assigned to the call by the CCF.
Refer to TS 129 002 for the actual definition of this parameter.

12.24   CallResult

This parameter provides the SCF with the charging related information previously requested using the "ApplyCharging" operation.
It comprises the following alternatives:

- callResult:
  The content of "CallResult" is network operator specific. This subparameter includes the "aChChargingAddress" parameter as received in the related "ApplyCharging" operation. Examples of these contents may be: bulk counter values, costs, tariff change and time of change, time stamps, duration, etc.; or

- callSupervisionResult:
  This subparameter contains the result of the requested call supervision in the ApplyCharging operation.
  It comprises the following fields:

  - callResult:
    This field provides the SCF with the charging related information previously requested with the aChBillingChargingCharacteristics using the "ApplyCharging" operation. The content of "CallResult" parameter is network operator specific. An example is correlation identification;

- reportConditionInformation:
  This field indicates the reason for reporting. Following indications are possible:

  - an intermediate report:
    indicates an intermediate ACR, reported at an interval as specified in the ApplyCharging operation
    parameter "reportCondition";

  - a final report when the call is released:
    indicates a final ACR reported when the credit limit is reached and the "connection" is released by the
    SSP, as specified in the ApplyCharging operation parameter releaseWhenLimitReached;

  - a final report when the call is not released:
    indicates a final ACR reported when the credit limit is reached and no instruction is received in the AC
    operation to release the "connection";

- aChChargingAddress:
  This parameter is the reference to the parameter "aChChargingAddress" as indicated in the "ApplyCharging"
  operation;

NOTE: The value of this parameter is not necessarily the same, i.e. after a "MoveCallSegments", operation.

- supervisionResult:
  This field reports back the used currency, pulses units or duration;

  This field comprises the following alternatives:

  - cost:
    This subfield contains the used cost. The measurement of the used cost is defined with the charging process
    applicable in the SSP. For example the FCI can specify a call attempt and/or a communication charge. It
    comprises the following elements:

    - usedCurrencyFactor:
      Indicates as value the total used currency between the moment of reporting and the start of the charging
      process or sending the ACR Final Report Call Active;

    - usedScale:
      Indicates the actually applied scale of the currency value in the charging process of the SSP at the
      moment of reporting;

    - reportedCurrency:
      Indicates the actually applied currency unit in the charging process of the SSP at the moment of reporting;
          or

  - pulses:
    The measurement of the used pulses is defined with the charging process applicable in the SSP. For example
    the FCI can specify a call attempt and/or a communication charge;
    It comprises the following element:

    - usedPulseUnits:
      This parameter contains the total used units of pulses between the moment of reporting and the start of the
      charging process or sending the ACR Final Report Call Active; or

  - time:
    indicates the used time for the actual call supervision applied;
    It comprises the following elements:

    - usedDuration:
      This parameter indicates the total used duration at the moment of reporting between either answer or the
      previously occurred tariff switch and either the sending of the "ApplyChargingReport" operation or the
      last occurred tariff switch;

    - usedDurationAfterSwitchOverTime:
      This parameter is only present if a tariff switch has occurred. This parameter indicates the total used
      duration at the moment of reporting between the occurred tariff switch and the sending of the
      "ApplyChargingReport" operation;

- minLimitNeeded:
  This element indicates the minimum limit required for successful continuation of the call. This parameter is set if e.g. a received addOnCharges cannot be handled with the old limit. It can be used by the SCF to calculate the new limit;

- extensions:
  Refer to EN 301 931-1 for a definition of this field.

12.25   CallSegment

For a definition of this parameter, refer to the 'CallSegmentID' clause.

12.26   CallSegmentID

This parameter contains an identifier that indicates the Call Segment (CS) to which the operation shall apply.

12.27   CallSegments

This parameter indicates the Call Segment in the source CSA and the equivalent Call Segment in the target CSA. It is used to change the logical identifier of a call segment (CSID) being moved from a source CSA into a new (target) CSA. It comprises the following sub-parameters:

- sourceCallSegment
  For a definition of this sub-parameter, refer to the 'CallSegmentID' clause.
  It is used to specify the source Call Segment identifier of the CS to be moved.

- newCallSegment
  For a definition of this sub-parameter, refer to the 'CallSegmentID' clause.
  It is used to specify the new Call Segment identifier of the moved CS.

12.28   CallSegmentToCancel

This parameter is defined in EN 301 931-3.

12.29   CallSegmentToRelease

This parameter indicates that a Call Segment shall be released. It comprises the following sub-parameters:

- callSegment
  For a definition of this sub-parameter, refer to the 'CallSegmentID' clause.

- cause
  For a definition of this sub-parameter, refer to the 'Cause' clause.

12.30   CallSupervision

This parameter indicates the methodology to apply call supervision for a specified aChChargingAddress, based on currency, pulse units or duration. It comprises the following subparameters:

-   supervisionMethod:
    This subparameter specifies a maximum cost, pulse units or duration given to supervise;
    It comprises the following alternatives:

    -   maximumTariffCurrency:
        This alternative specifies the maximum amount of currency at which a communication can be maintained. It comprises the following fields:

        -   currencyFactor:
            This field indicates the maximum tariff currency value in units;

        -   currencyScale:
            This field provides a currency scale to apply to the given maximum value. When not specified the default value is "noScale";

        -   currency:
            This field provides the currency unit to apply to the maximum given value. If not specified, the default is "noIndication"; or

    -   maximumTimeDuration:
        This alternative specifies the maximum time a communication can be maintained. It comprises the following fields:

        -   maximumDurationAllowed:
            This field indicates the maximum duration allowed for the communication;

        -   switchOverTime:
            This field indicates the tariff switch applied in the SCP to be taken into account by the SSF in reporting the used time before and after this switch over time. The measurement in the SSF of this switch over time is started at receiving this parameter in the AC operation. No correlation is applicable between this parameter and possible other tariff switches applied in the SSP; or

    -   maximumPulseUnits:
        This alternative specifies the maximum number of pulses that can be applied for the communication;

-   warningBeforeLimitReached:
    This subparameter indicates the warning to be given before the credit limit is reached of the actual call supervision in progress. In case this parameter is not present, no warning shall be given.
    It comprises the following fields:

    -   durationBeforeLimitReached:
        This field indicates the duration before the limit, the maximum cost, pulse units or duration is reached to send the warning;

    -   warningToSend:
        This field indicates the warning tone to e applied. If this parameter is not present, a default tone shall be applied by the SSP;

    -   warningDirection:
        This field shall indicate the legID to where the warning tone has to be transmitted. In case this parameter is not present, the default is that the warning before limit is sent to all joined legs in the call segment associated to the aChChargingAddress;

- releaseWhenLimitReached:
  This subparameter indicates in case when the call supervision limit is reached, the release cause to be applied by the SSF to release the connection. In case not specified the default release cause (normal, unspecified) shall be applied. In case this parameter is not present, no release shall be applied by the SSP.
  It comprises the following field:

  - releaseCause:
    For a definition of this field see clause 'ReleaseCause' for a definition;

- reportConditions:
  In case this parameter is not present only one final report is requested.
  It comprises the following field:

  - periodicReportinterval:
    This field indicates the interval in seconds to send the intermediate ApplyChargingReport operations to the SCF. The interval starts either with the start of the time measurement respectively start of the charging or at receipt of the new AC if it has already started.

12.31   CAMEL-AChBillingChargingCharacteristics

This parameter specifies a list of sub-parameters required for CSE control of call duration.

The list may contain:

- timeDurationCharging:
  This list contains the following parameters:

  - maxCallPeriodDuration:
    This parameter specifies the period of time for which a call can progress before an ApplyChargingReport shall be sent to the SCF;

  - releaseIfdurationExceeded:
    This parameter specifies the action to be taken at the SSF when the duration specified above has been reached. If the parameter is present, then the call is released;

- Tone:
  If the parameter is present, then a warning tone is played when the warning tone timer expires;

- tariffSwitchInterval:
  This parameter indicates to the SSF the time duration until the next tariff switch. The measurement of the elapsed tariff switch period commences immediately upon successful execution of this operation.

12.32   CAMEL-CallResult

This parameter provides the SCF with the charging related information previously requested using the ApplyCharging operation. The "CallResult" is a list, and can contain the following parameters:

- timeDurationChargingResult:
  This is a list, and can contain the following parameters:

- timeInformation:
  This is a choice of the following parameters:

  - timeIfNoTariffSwitch:
    This parameter will be present if no tariff switch has occurred since the detection of Answer for the connection to the Called Party, Temporary Connection or SRF connection, otherwise it will be absent. If present, then the elapsed time since detection of Answer is reported;

- timeIfTariffSwitch:
This parameter will be present if a tariff switch has occurred since the detection of Answer for the connection to the Called Party, Temporary Connection or SRF connection, otherwise it will be absent. If present, then the parameter may contain the following information:

  - timeSinceLastTariffSwitch:
  The elapsed time since detection of the last tariff switch is reported;

  - tariffSwitchInterval:
  This parameter is present only if a tariff switch was detected for the connection to the Called Party, the temporary connection or the SRF connection within the reported call period.
  If present the time interval between either the detection of the Answer event or the previous tariff switch (whichever of these events was last detected) and the last tariff switch is reported.

- partyToCharge:
The "partyToCharge" parameter as received in the related ApplyCharging operation or deduced from the default value, to correlate the result to the request;

- CallActive:
This parameter indicates whether the call is still active or has been released;

- CallReleasedAtTcpExpiry:
This present indicates that the SSF has released the call and terminated the dialogue.
It shall be present when ACR is sent due to Tcp expiry and the SSF has released the call (because ReleaseIfExceeded was present in ACH) and terminated the dialogue.
In all other instances, this parameter shall be absent.

12.33   CAMEL-FCIBillingChargingCharacteristics

This parameter contains the following sub-parameters:

- FCIBCCCAMELsequence1:
This parameter contains the following sub-parameters;

- FreeFormatData:
This parameter contains free-format billing and/or charging characteristics;

- PartyToCharge:
This parameter indicates the party to bill and/or charge.

12.34   CAMEL-SCIBillingChargingCharacteristics

This parameter is a choice between two lists of information.

The first list shall only be sent before an answer event has been detected from the current Called Party, Temporary Connection or connection to an SRF. It contains the following parameters:

- aOCBeforeAnswer:
This is a list of the following information:

  - aOCInitial:
  This is a set of GSM Charge Advice Information elements, as defined in TS 122 024, and these CAI elements are sent by the SSF to the MS when an ANSWER is received and a tariff switch has not yet occurred;

  - aOCSubsequent:
  This list may indicate the following information:

    - CAIElements:
    This is a set of GSM Charge Advice Information elements, as defined in TS 122 024, and these CAI elements are sent to the MS when Answer is detected and a tariff switch has occurred previously, or when Answer has previously been detected and a tariff switch occurs;

- tariffSwitchInterval:
This parameter indicates to the SSF the time duration until the next tariff switch. The measurement of the elapsed tariff switch period commences immediately upon successful execution of this operation.

The second list in the Choice shall only be sent after an answer event has been detected from the current Called Party, Temporary Connection or connection to an SRF. It contains the following parameters:

- aOCAfterAnswer:
This list may indicate the following information:

    - cAIElements:
    This is a set of GSM Charge Advice Information elements, as defined in TS 122 024, and these CAI elements are sent to the MS by the SSF when Answer is detected and a tariff switch has occurred previously, or when Answer has previously been detected and a tariff switch occurs in the call;

    - tariffSwitchInterval:
    This parameter indicates to the SSF the time duration until the next tariff switch. The measurement of the elapsed tariff switch period commences immediately upon successful execution of this operation;

    - legID:
    This parameter indicates where the charging information shall be sent. For Mobile Originated calls, only leg 1 shall be used. For Mobile Terminated calls in the VMSC, only leg 2 shall be used.

12.35   Carrier

This parameter indicates carrier information. It consists of the carrier selection field followed by either the Carrier ID information or Transit Network Selection information to be used by SSF for routing a call to a carrier.

It comprises the following **embedded** field:

- carrierSelectionField:
This field indicates how the selected carrier is provided (e.g. pre-subscribed).

It also comprises the following alternative **embedded** fields:

- carrierID:
This alternative indicates the carrier to use for the call. It contains the digits of the carrier identification code. Refer to ITU-T Recommendation Q.1290 for the actual definition of carrier identification code; or

- TransitNetworkSelection
This alternative indicates the carrier to use for the call. It contains the Transit Network Selection information, which indicates the transit network(s), requested to be used in the call. Refer to ITU-T Recommendation Q.762 for the actual definition of this field.

NOTE: Which alternative should be used is dependent on the network. It is a hard-coded decision based on the region (e.g. option 1 in North America (na) and option 2 in Europe) in which the switch is located.

For CAMEL the na CarrierInformation is applied: the carrier selection field is equivalent to the na CICSelectionType and the second field carrierID is equivalent to the na carrierID. The na CarrierInformation is included at the discretion of the gsmSSF operator.

12.36   Cause

This parameter contains a number giving an indication to the SSF or the SCF about the reason of a signal e.g. at releasing a call. This may be used by SSF for generating specific tones to the different parties in the call or to fill in the "cause" in the release message. For the SCF it may be used to identify the cause for a leg being released (e.g. "a ported subscriber number" or "called party not reachable", etc.). Refer to ITU-T Recommendation Q.850 for the use of cause and location values and to ITU-T Recommendation Q.762 'cause indicators' for the actual definition of this parameter.

12.37   CCSS

This parameter CCSS (Call Completion on Service Set-up) indicates to the SCF that the current call is due to a special procedure (CCBS or CCNR).

12.38   CGEncountered

This parameter indicates the type of call gapping which the request for instructions has been subjected to, if any.

12.39   ChargeNumber

This parameter contains the number that identifies the entity to be charged for the call.
It identifies the chargeable number for the usage of a carrier (applicable on a call sent into a North American long distance carrier). For a definition of this parameter as used by CAMEL refer to ANSI ISUP T1.113.

12.40   ChargingControl

This parameter contains charging control information.
It comprises the sub-parameters tariffInfoFromSuccExchangeSCI and sSFdetermination.

- tariffInfoFromSuccExchangeSCI:
   This subparameter indicates how charge messages received from succeeding exchange are to be handled in the SSF. It comprises the following fields.
   There are two separate sub-fields defined. Sub-parameter relayChargesFromDestination and sub-parameter relayAOCfromDestination:

   - relayChargesFromDestination:
      This field controls the relay of charge messages which are related to the subscriber charge.
      The following values are defined:

      - noIndication:
         This value indicates, that not the SCF but the SSF has to decide, whether charge messages received from B-side are to be forwarded to the partyToCharge or not. Therefore the parameter sSFdetermination shall indicate one of the values destinationRoutingAddress or calledInNumber. In case that the parameter sSFdetermination is not present and the default value noDetermination applies or if the parameter is present and has the value noDetermination, charge messages received from succeeding exchange are not to be relayed to the partyToCharge;

      - relay:
         This value indicates that all charge messages received from B-side are to be forwarded to the partyToCharge;

      - noRelay:
         This value indicates that charge messages received from B-side are not to be forwarded to the partyToCharge;

   - relayAOCfromDestination:
      This field controls the relay of charge messages which are related to the 'AOC only'.
      It indicates that all charge messages with indication 'AOC only' received from B-side are to be forwarded to the partyToCharge if it is set to TRUE. Otherwise, if it is set to FALSE, charge messages with indication 'AOC only' received from B-side are not to be forwarded to the partyToCharge;

- sSFdetermination:
   This subparameter indicates, if the SSP (CCF) has to determine the charges and the handling for charge messages from B-side by means of its own database. Following values are possible:

   - noDetermination: No determination has to be done at the SSP (CCF);

- destinationRoutingAddress: Determination based on the destination routing address digits shall be done. The destination routing address digits are the digits which are used by the SSP to establish a connection to a B – subscriber;

- calledInNumber: Determination based on the called IN number shall be done. The called IN Number is the calledPartyNumber in the InitialDP operation.

12.41   ChargingEvents

This parameter contains a list (1 to n) of charging events and the corresponding monitor type and corresponding leg. For each chargingEvent in the list it comprises the following sub-parameters:

- eventTypeCharging:
  This sub-parameter indicates the charging event type. Its content is network operator specific, which may be "charge pulses" or "charge messages";

- monitorMode:
  This sub parameter indicates the monitorMode applicable for the corresponding "eventTypeCharging" or "eventTypeTariff" sub parameter. Monitor may be "interrupted", "notifyAndContinue" or "transparent";

- legID:
  For a definition of this sub-parameter, refer to the 'LegID' clause;

- eventTypeTariff:
  This subparameter indicates the charging event type related to tariff determination. The charging event may include information about tariff-, add-on charge-, tariff/add-on charge acknowledgement -, or tariff/add-on charge-acknowledgement timer expiry.

12.42   CNinfo

This parameter includes VPN call information to identify the Corporate telecommunication Network. "VPN" refers to a Virtual Private Network with the support of PSS1 information flows.
It contains the Corporate Telecommunications Network Identifier (CNID) Indicator followed by the Corporate Telecommunications Network Identifier, if present.
Refer to ITU-T Recommendation Q.765.1 for a definition of this parameter.

12.43   ConnectTime

This parameter indicates the time duration a leg is connected.
For a called party leg it is the time between the received answer indication from the indicated or default called party side for that connection or party; and

a) the detection of a release event on that leg (connect time may e.g. be reported in ODisconnect, TDisconnect, EventReportBCSM); or

b) the detection of the required midcall event on that leg (connect time may e.g. be reported in OMidCall, TMidCall, EventReportBCSM).

For a calling party leg it is the time between the sending of InitialDP; and

a) the detection of a release event (ODisconnect, TDisconnect, EventReportBCSM) of that party (i.e. leg) to be reported; or

b) the sending of InitialDP and the detection of the mid call event (EventReportBCSM, OMidCall, TMidCall) to be reported.

12.44　ControlType

This parameter indicates the reason for activating call gapping. The following values are defined:

- The "controlType" value '*sCPOverloaded*' indicates that an automatic congestion detection and control mechanism in the SCP has detected a congestion situation.

- The "controlType" value '*manuallyInitiated*' indicates that the service and or network/service management centre has detected a congestion situation, or any other situation that requires manually initiated controls.

- The "controlType" value '*destinationOverload*' indicates that the destination entity (e.g. SRF) has detected a congestion situation.

12.45　CorrelationID

This parameter contains an identifier used by the SCF for correlation with a previous operation.
It is for example used by the SCF to a) associate the "AssistRequestInstructions" operation from the assisting SSF with the Request from the initiating SSF or b) to associate the "StatusReport" from the SSF with the request.
The network operator has to decide about the actual mapping of this parameter on the used signalling system.

12.46　CountersValue

This parameter specifies a list of (1 to n) requested e count of calls filtered during the filtering period. For each count, it comprises the following sub-parameters:

- counterID:
  This sub-parameter identifies the counter;

- counterValue:
  This sub-parameter contains the related counter value.

12.47　CreatedCallSegmentAssociation

This parameter identifies for the SCF unambiguously the CSA instance in the SSF under SCF control. This CSA identifier assigned by the SSF may be used to associate different CSA instances in the SSF. It should exist, when the SSF has not previously informed the created CSA Identifier to the SCF, i.e. in case of a TDP.

12.48　CreateOrRemove

This parameter indicates if a new trigger is to be created or an existing trigger is to be removed.
In case the createOrRemove indicator is set to « remove, only the TriggerIdentifier is included (and optionally the registratorIdentifier). The TriggerIdentifier is not included when the createOrRemove indicator s set to « create ».

12.49　CSFailure

This parameter indicates that a call segment has been released due to a failure. It comprises the following sub-parameters:

- callSegmentID:
  For a definition of this sub-parameter, refer to the 'CallSegmentID' clause;

- reason:
  This sub-parameter provides optional network specific information about the kind of error/exception (e.g. external or internal error or exception). Its content is network operator specific;

- cause:
  For a definition of this sub-parameter, refer to the 'Cause' clause.

12.50   Cug-Index

This parameter is used to select a CUG for an outgoing call at the user, or to indicate an incoming CUG call to the user.

12.51   Cug-Interlock

This parameter uniquely identifies a CUG within a network.

12.52   Cug-OutgoingAccess

This parameter indicates if the calling user has subscribed to the outgoing access inter-CUG accessibility subscription option.

12.53   CutAndPaste

This parameter is used by the SCF to instruct the SSF to delete (cut) a specified number of leading digits that it has received from the calling party and to paste the remaining dialled digits on to the end of the digits supplied by the SCF in the 'destinationRoutingAddress' parameter".

12.54   DefaultFaultHandling

This parameter indicates which default treatment shall be applied to the call, which encounter a trigger (TDP) in case the SCF is not reachable. It comprises the following sub-parameters:

- action:
  This sub-parameter indicates the action to be taken. The following alternatives have been defined:

  - *resume call processing:* call processing shall be resumed as if a Continue/ContinueWithArgument operation had been received;

  - *release call:* the call shall be released with a locally defined cause value;

- treatment:
  This sub-parameter specifies a possible treatment (e.g. sending of announcement before release) in relation to the selected action. For a definition of this sub-parameter refer to the 'GapTreatment' clause.

12.55   DestinationNumberRoutingAddress

This parameter contains the called party number towards which the call is to be routed.
Refer to ITU-T Recommendation Q.762 'called party number' for the actual definition of this parameter.

12.56   DestinationRoutingAddress

This parameter specifies a Called Party Number. It comprises the following sub-parameter:

- calledPartyNumber:
  For a definition of this sub-parameter, refer to the 'CalledPartyNumber' clause.

12.57   DisplayInformation

This parameter indicates a text string to be sent to the end user.
This information cannot be received by a PSTN end-user. Refer to ITU-T Recommendation Q.762 'display information' for the actual definition of this parameter.

Delivery of DisplayInformation parameter to Private Networks cannot be guaranteed.
Refer to ITU-T Recommendation Q.1601 regarding interworking.

12.58   DPName

This parameter identifies the name of the DP in the O_BCSM respective T-BCSM. It contains an indication for the BCSM DP event.

12.59   EventSpecificInformationCharging

This parameter contains charging related information specific to the event. Its content is network operator specific.

12.60   EventSpecificInformationBCSM

This parameter indicates the call related information specific to the event. It comprises the following alternatives:

- collectedInfoSpecificInfo
  This alternative specifies the collected information, it contains a number including all collected digits.
  It comprises the following field:

  - calledPartynumber
    This field contains all available collected digits in the called number, it may be an incomplete number; or

- analysedInfoSpecificInfo
  This alternative contains the analysed information. It comprises the following field:

  - calledPartynumber
    This field contains the analysed called number, it may be an incomplete number; or

- routeSelectFailureSpecificInfo;

- failureCause
  For a definition of this field, refer to the 'FailureCause' clause; or

- oCalledPartyBusySpecificInfo;

- busyCause
  For a definition of this field, refer to the 'BusyCause' clause; or

- oNoAnswerSpecificInfo;

- oNoAnswerCause
  For a definition of this field, refer to the 'Cause' clause; or

- oAnswerSpecificInfo;

- backwardGVNS
  For a definition of this field, refer to the 'BackwardGVNS' clause;

- destinationAddress
  The destination address for the call;

- or-Call
  The OR indicator if the call was subject to basic optimal routeing as specified in TS 123 079;

- forwardedCall
  The forwarding indicator if the Call Forwarding Supplementary Service was invoked; or

- oMidCallSpecificInfo;

- connectTime
  For a definition of this field, refer to the 'ConnectTime' clause;

- oMidCallInfo
  This field contains the digits representing IN service control code of the midcall event detected; or

- oDisconnectSpecificInfo;

- releaseCause
  For a definition of this field, refer to the 'ReleaseCause' clause;

- connectTime
  For a definition of this field, refer to the 'ConnectTime' clause; or

- tBusySpecificInfo;

- busyCause
  For a definition of this field, refer to the 'BusyCause' clause;

- callForwarded
  If the T-busy event is triggered by call forwarding at the GMSC/VMSC, the eventSpecificInformationBCSM
  will contain the CallForwarded indication; or

- tNoAnswerSpecificInfo;

- tNoAnswerCause
  For a definition of this field, refer to the 'Cause' clause;

- callForwarded
  If the no answer event is triggered by call forwarding at the GMSC/VMSC; the eventSpecificInformationBCSM
  will contain the CallForwarded indication; or

- tAnswerSpecificInfo;

- destinationAddress
  The destination address for the call;

- or-Call
  The OR indicator if the call was subject to basic optimal routeing as specified in TS 123 079;

- forwardedCall
  The forwarding indicator if the Call Forwarding Supplementary Service was invoked; or

- tMidCallSpecificInfo;

- connectTime
  For a definition of this field, refer to the 'ConnectTime' clause;

- tMidCallInfo
  This field contains the digits representing IN service control code of the midcall event detected; or

- tDisconnectSpecificInfo;

- releaseCause
  For a definition of this field, refer to the 'ReleaseCause' clause;

- connectTime
  For a definition of this field, refer to the 'ConnectTime' clause; or

- oTermSeizedSpecificInfo
  - no specific info defined; or

- oSuspend
  - no specific info defined; or

- tSuspend
  - no specific info defined; or

- origAttemptAuthorized
  - no specific info defined; or

- oReAnswer
  - no specific info defined; or

- tReAnswer
  - no specific info defined; or

- facilitySelectedAndAvailable
  - no specific info defined; or

- callAccepted
  - no specific info defined; or

- oAbandon;

- abandonCause
  For a definition of this field, refer to the 'Cause' clause; or

- tAbandon;

- abandonCause
  For a definition of this field, refer to the 'Cause' clause; or

- authorizeRouteFailure;

- authorizeRouteFailureCause
  For a definition of this field, refer to the 'Cause' clause; or

- terminationAttemptAuthorized
  - no specific info defined; or

- originationAttemptDenied;

- originationDeniedCause
  For a definition of this field, refer to the 'Cause' clause; or

- terminationAttemptDenied;

- terminationDeniedCause
  For a definition of this field, refer to the 'Cause' clause.

12.61   EventSpecificInformationTariff

This parameter contains charging tariff related information specific to the event. It may include the ChargingTariffInformation -, AddOnChargingInformation -, ChargingAcknowledgementInformation -, StartCharging or StopCharging sub-parameter.

12.62   EventTypeBCSM

This parameter is identical to the 'DPName' parameter. For a definition of this parameter, refer to the 'DPName' clause.

12.63 EventTypeCharging

This parameter indicates the charging event type which has occurred. Its content is network operator specific, which may be "charge pulses" or "charge messages".

12.64 EventTypeTariff

This parameter indicates the charging event type related to tariff determination. This charging event may include information about tariff -, add-on charge -, start charge -, stop charge, acknowledgement -, or -acknowledgement timer expiry.

12.65 Ext-BasicServiceCode

Indicates the Basic Service Code. Refer to TS 129 002 for a definition of this parameter.

12.66 Extensions

This parameter is defined in EN 301 931-1.

12.67 FailureCause

For a definition of this parameter, refer to the 'Cause' clause. It is used to contain a cause value that specifies why the call failed e.g. route failure.

12.68 FCIBillingChargingCharacteristics

This parameter indicates billing and/or charging characteristics. For CS1 and CS2 respectively it consists of network operator specific parts and a part for the determination of tariff/price.
In addition this parameter consist of a part for generation of a logical call data record, a part for including information in the logical call data record and a part that activates and influences the charging in the SSF.

It comprises the following alternatives:

- fCIBCCcs1:
  This subparameter is network operator specific and replaces the fCIBillingChargingCharacteristics for CS1; or

- fCIBCCsequence:
  This subparameter consists of the following fields:

  - fCIBCC:
    The content of this field is network operator specific;

  - chargeMessageFromSCF:
    This field comprises the following alternatives:

    - sCFChargingTariff
      This field contains a tariff to be sent to the charging instance in the SSF in order to initiate or change the IN charging for the specified leg; or

    - sCFAddOncharge:
      This field contains a number of units to be added to the charging instance in the SSF for the specified chargingAddress.

  NOTE: The reception of this field in the SSF will not cause any signalling in the network.

Following sub-parameters are not relevant:

- sub-parameters subscriberCharge and delayUntilStart within parameter chargingControlIndicators;

- parameters originationIdentification and destinationIdentification.

If received they should be ignored by the SSF.

- tariffFromSuccExchange:
  This field is used to indicate whether charging message(s) received from a succeeding exchange shall be applied in the IN charging in the SSF and whether this message if it is to be used for advice of charge should be further propagated. The charging message will contain either ChargingTariffInformation or AddOnChargingInformation. The indicator has the following values:

  - *notTakenIntoAccount:*
    Charge messages received from a succeeding exchange shall not be applied in IN charging in the SSF;

  - *takeIntoAccountNoAOC:*
    The SSF shall do IN charging in the SSF according to charge messages received from succeeding exchanges. Charge messages that also could be used for advice of charge are not further propagated;

  - *takeIntoAcountTranslateIntoAOC:*
    The SSF shall do IN charging in the SSF according to charge messages received from succeeding exchanges. Charge messages that also could be used for advice of charge are further propagated;

- freeFormatData:
  This field contains additional information, to be written into the logical call data record for off-line charging purposes, the information can be given one or several times;

- chargingAddress:
  This field indicates the charging address, legID or srfCallSegmentID, to which the "FurnishCharging" operation shall be applied. If not present, then it is applied to the default legID 2.
  This field comprises the following alternatives:

  - leg:
    The charging is associated to a specified leg; or

  - srfCallSegment:
    The charging is associated to a specified call segment. This field is used to indicate that the charging is related to a User Interaction SRF connection for a call segment.

- partyToChargeIdentifier:
  This field contains information to be written into the logical call data record, identifying the party to which the IN charging applies. The party identifiers may e.g. indicate a subscriber in the call, a third party or a service provider. The partyToChargeIdentifier comprises the following fields:

  - chargedParty:
    Indicates who has to be charged;

  - specialINNumber:
    An optional identifier that represents the subscriber or entity to which the IN charging applies;

- iNRecordIndicators:
  This field is used to indicate whether a logical call data record must be generated and how data must be written to the logical call data record. In addition to the iNRecordIndicator tariffs information for charge supervision may be delivered where the FCI is used together with the AC/ACR surveillance of a maximum currency or pulse value.
  The iNRecordIndicators consist of the following fields:

  - iNRecordAction:
    It comprises the following values:

    - *GenerateCallRecord*
      generate a logical call data record and append data (freeFormatData and partyToChargeIdentifier) to the record. Any previous logical call data record for the same chargingAddress shall be closed. The closing of the existing logical call data record generates an output. This output can be complete (not partial) or partial as defined in I-ETS 300 819;

- *AppendDataToCallRecord*
append data (freeFormatData and partyToChargeIdentifier) to existing logical call data record;

- *OverwriteDataForCallRecord*
overwrite existing data (freeFormatData and partyToChargeIdentifier) to existing logical call data record;

- *NoCallRecordInformation*
no data for logical call data record. In this case the FCI is used in connection with charge supervision together with the AC/ACR surveillance of a maximum currency or pulse value. This parameter can also be used to update charging tariff information or to send add-on charging information;

- hotBillingRequired
Indicates whether Hot Billing is required or not for the logical call data record mapped to a specific IN call data record. If Hot Billing is required real-time processing of the logical Billing record must be done. The real-time processing of the logical call data record will be operator specific. The request for Hot-Billing is only applicable if iNRecordAction is GenerateCallRecord.

12.69   FilteredCallTreatment

This parameter specifies how filtered calls are treated. It includes information about the announcement to be played the charging approach, the number of counters used and the release cause to be applied to filtered calls.
It comprises the following sub-parameters:

- sFBillingChargingCharacteristics
This parameter determines the charging to be applied for service filtering. Its content is network specific;

- informationToSend
This sub-parameter is defined in EN 301 931-3;

- maximumNumberOfCounters
For a definition of this sub-parameter, refer to the 'MaximumNumberOfCounters' clause;

- releaseCause
For a definition of this sub-parameter, refer to the 'ReleaseCause' clause;

- sFTariffMessage
This sub-parameter determines the tariff to be applied to the filtered call. It may include the crgt sub-parameter but only for subscriber charge. The charge specified in the tariff can either be Call setup charge or Call attempt charge.

12.70   FilteringCharacteristics

This parameter indicates the severity of the filtering and the point in time when the "ServiceFilteringResponse" shall be sent. It comprises the following alternatives:

- interval
This alternative indicates that for every interval of time the next call leads to an InitialDP and a ServiceFilteringResponse is sent to the SCF. This sub-parameter contains the value for the interval time.
An interval of 0 indicates that all calls matching the filtering criteria will result in sending of an "InitialDP" operation and no filtering will be applied (i.e. no "ServiceFilteringResponse will be sent).
An interval of -1 indicates that none of the calls matching the filtering criteria will either result in sending of an "InitialDP" or a "ServiceFilteringResponse" operation
Other values of Interval indicate duration in seconds; or

- numberOfCalls
This alternative indicates that for every N calls the Nth call shall lead to an InitialDP and a ServiceFilteringResponse. This sub-parameter contains the value N for number of calls.
If ActivateServiceFiltering implies several counters - filtering on several dialled numbers -, the numberOfCalls would include calls to all the dialled numbers.

12.71   FilteringCriteria

This parameter indicates which calls are filtered based on specified criteria.
It comprises the following alternatives:

- serviceKey

    For a definition of this alternative, refer to the 'ServiceKey' clause
    This alternative identifies unambiguously the requested IN service for which filtering should be applied; or

- addressAndService

    This alternative identifies the IN service and dialled number for which filtering should be applied. The geographical area may also be identified ("callingAddressValue" and/or "locationNumber").
    This alternative comprises the following fields:

    - calledAddressValue
      This field identifies the number (i.e. digits) of the called party;

    - serviceKey
      For a definition of this alternative, refer to the 'ServiceKey' clause;

    - callingAddressValue
      This field identifies the number (i.e. digits) of the calling party;

    - locationNumber
      For a definition of this alternative, refer to the 'LocationNumber' clause.

12.72   FilteringTimeOut

This parameter indicates the duration of the filtering. It comprises the following alternatives:

- duration

    This alternative indicates the duration time for service filtering being active or the immediate request to stop (zero as duration time indicates removal) an already active service filtering.
    Duration of 0 indicates that service filtering is to be removed.
    Duration of -1 indicates an infinite duration.
    Duration of -2 indicates a network specific duration.
    Other values indicate duration in seconds;

- stopTime

    This alternative indicates the time (date and time) when the active service filtering is to be ended (i.e. removed) or the immediate stop, i.e. the request to remove an already activated service filtering.

12.73   ForcedRelease

This parameter indicates that a forced release is requested, i.e. a release in any state regardless of the CCF/SSF state (except Idle).
If an SLP wants to ensure that a call will be released immediately on receipt of the ReleaseCall operation, regardless of the state of the **FSM for CS** in the SSF, this parameter shall be set to "TRUE ". The forced release applies for the requested call segment(s) within the Call Segment Association.
If this parameter is omitted (default value is "FALSE") the default release procedure applies, i.e. release procedure shall be in accordance to the **FSM for CS** state.

12.74   ForwardCallIndicators

This parameter indicates if the call shall be treated as a national or international call. It also indicates the signalling capabilities of the network access, preceding network connection and the preferred signalling capabilities of the succeeding network connection. The network access capabilities does not indicate the terminal type. For example, an ISPBX will have an ISDN type of access, but the end user terminal behind the ISPBX may be ISDN or non-ISDN.

12.75   ForwardGVNS

This parameter identifies the originating service provider and provides information about the calling VPN user in terms of a customerID or a GVNS user group. The parameter will also carry routing information for the terminating GVNS network. Refer to ITU-T Recommendation Q.762 for the actual definition of this parameter.

12.76   GapCriteria

This parameter identifies the criteria for a call to be subject to service request gapping. It comprises the following alternatives:

> basicGapCriteria; or

> compoundGapCriteria
> This parameter comprises the following subparameters:

> -   basicGapCriteria;

> -   scfID
>   This subparameter is restricted to include a fixed GT address string.

NOTE 1:   In the case where the GT addresses more than one SCP (e.g. a mated pair) then
if one of these physical SCPs enters overload conditions and issues CallGap, then it is
applied to all of them.

For a definition of this subparameter, refer to the 'ScfID' clause.

The basicGapCriteria comprises the following subparameters:

-   calledAddressValue
    This sub-parameter indicates that service request gapping will be applied when the leading digits of the dialled number of a call attempt match those specified in "gapCriteria"; or

-   gapOnService
    This sub-parameter indicates that service request gapping will be applied when the "servicekey" of a call attempt match those specified in "gapCriteria"; or

-   calledAddressAndService
    This sub-parameter indicates that service request gapping will be applied when the "serviceKey" and the leading digits of the dialled number of a call attempt match those specified in "gapCriteria".
    It comprises the following fields:

    -   calledAddressValue;

    -   serviceKey; or

-   callingAddressAndService
    This sub-parameter indicates that service request gapping will be applied when the "serviceKey" and the leading digits of the calling party number or the location number of a call attempt match those specified in "gapCriteria".
    It comprises the following fields:

    -   callingAddressValue
        This field contains the leading digits of either the calling party number or the location number;

    -   serviceKey
        For a definition of this field, refer to the 'ServiceKey' clause;

- locationNumber
For a definition of this field, refer to the 'LocationNumber' clause; or

- gapAllINTraffic
This sub-parameter indicates that service request gapping will be applied for every IN call.

NOTE 2: The use of this parameter to gap traffic to one SCP may gap the traffic to other SCPs.
Solutions to this problem are for to be resolved in future Capability Set.

compoundGapCriteria

12.77   GapIndicators

This parameter indicates the gapping characteristics. It comprises the following sub-parameters:

- duration
This sub-parameter specifies the total time interval during which service requests gapping for the specified gap
criteria will be active.
Duration of 0 indicates that gapping is to be removed.
Duration of -1 indicates an infinite duration.
Duration of -2 indicates a network specific duration.
Other values indicate duration (in seconds).

- gapInterval
This sub-parameter specifies the minimum time between service requests being allowed through.

An interval of 0 indicates that service requests meeting the gap criteria are not
to be rejected, i.e. no gapping when gapInterval equals 0; and
an interval of -1 indicates that all service requests meeting the gap criteria are
to be rejected, i.e.gap all calls when gapInterval equals -1.

Other values indicate interval (in milliseconds).

12.78   GapTreatment

This parameter indicates how calls that were stopped by the service request gapping mechanism shall be treated.
It comprises the following alternative sub-parameters.

- informationToSend
This sub-parameter is defined in EN 301 931-3.
This alternative is used to specify the information to be sent to the calling party. At the end of the information
sending the call shall be released, with a locally defined cause, unless a specific default treatment depending on
network operator is selected. The default treatment can e.g. be to release the call or to resume call processing; or

- releaseCause
For a definition of this parameter, refer to the 'Cause' clause.
This alternative is used to indicate that the call shall be released and specifies the cause to be send to the calling
party; or

- both
This alternative indicates both the information to be sent to the calling party and the cause to be used when
releasing the call at the end of the information sending. It comprises the following fields:

    - informationToSend
    This field is defined in EN 301 931-3.
    It is used to specify the information to be sent to the calling party. At the end of the information sending the
    call shall be released, with a locally defined cause, unless a specific default treatment depending on network
    operator is selected. The default treatment can e.g. be to release the call or to resume call processing;

- releaseCause
  For a definition of this field, refer to the 'ReleaseCause' clause.
  This field is used to indicate that the call shall be released and specifies the cause to be sent to the calling party.

12.79   GenericName

This parameter indicates the Call Party Name to be displayed to the end-user.

12.80   GenericNumbers

This parameter contains Number information sent to enhance network operation or for supplementary services
It allows the SCF to modify the GenericNumber information received from the CCF/SSF, if any. Also, it allows the SCF to precise a Generic Number information to the CCF/SSF.

This parameter allows the SCF to set the Generic Number parameter used in the network. It may be used for transfer of Additional Calling Party Number.

The SSF may inform the SCF about e.g. the additional calling party number but also with the called number, the additional connected number, the additional original called party number, the additional redirecting number and/or the additional redirection number. Refer to ITU-T Recommendation Q.762 for the actual definition of this parameter.

12.81   GlobalCallReference

This parameter contains a unique global identity of the call. Its purpose is to assist the offline correlation of Charging Data Records (CDR) for the same call by using a common call reference which uniquely identifies the call. This reference provided by the CCF could be copied into a SCP generated CDR for the call.
It comprises the following sub-parameters:

- networkID
  This subparameter specifies the identity of the operator for an incoming call.
  In the SSP that has the role of "Gateway" between operator networks this parameter does not have to be received by signalling. This type of SSP can identify this operator identity by the incoming trunk.
  It includes both the identification of the network and the identification of the node;

- callReferenceID
  This subparameter contains a binary number used for the call reference of the call. This is generated by the node for each call.

The value is unique within one network and the time period before reuse of the value will be network dependent. When transit through a private network the uniqueness of the call reference identifier parameter is not maintained".

The callReference ID length indicator can be put to zero in the globalCallReference parameter,
so that the network ID can be used without any callreference ID.

In the ISUP the Call Reference and the Global Call Reference parameters will never be passed simultaneously.

Refer to annex E.

12.82   GmscAddress

This parameter gives the gmscId assigned to the GMSC. For encoding see TS 129 002.

12.83   Gsm-ForwardingPending

This parameter indicates that a forwarded-to-number was received and the call will be forwarded due to GSM supplementary service call forwarding in the GMSC.

12.84   HighlayerCompatibility

This parameter indicates the type of the high layer compatibility, which will be used to determine the ISDN - teleservice of a connected ISDN terminal. For encoding DSS1 EN 300 403-1 is used. The highlayerCompatibility can also be transported by ISUP within the ATP (see ITU-T Recommendation Q.763) parameter.

12.85   IMSI

IMSI of the mobile subscriber for which the service is invoked. For encoding see TS 129 002.

12.86   InitialCallSegment

This parameter identifies the initial Call Segment. The initial Call Segment represents the first call segment that was created when the CSA was created, i.e. the CS where the trigger took place on a half-call or the CS that was created by an InitiateCallAttempt as a new control relation was established by the SCF.

12.87   InProfiles

This parameter specifies a list (1 to n) of INProfile identifying the triggers to be managed in the CCF/SSF and the associated action to be applied. For each INProfile it comprises the following sub-parameters:

- tDPIdentifier
  For a definition of this sub-parameter, refer to the 'TDPIdentifier' clause;

- dPName
  For a definition of this sub-parameter, refer to the 'DPName' clause;

- actionOnProfile
  This sub-parameter indicates the action to be performed i.e. *activate or- de-activate* the indicated trigger(s);

- extensions
  This sub-parameter is defined in EN 301 931-1.

12.88   INServiceCompatibilityIndication

This parameter contains the identifier for a class of IN services/service features that have been triggered during the call. A class of IN services is defined as IN services/service features which have the same compatibility characteristics.

12.89   INServiceCompatibilityResponse

This parameter is used by the SCF to inform the SSF about the actual services/service features, which have been invoked in the SCF. It is used by the SSF to overwrite the INServiceCompatibilityIndication which has been derived during triggering of the given IN service. It is up to the Network Operator whether overwrite is allowed or not.

12.90   InvokeID

This parameter is defined in EN 301 931-3.

12.91   IPAvailable

This parameter indicates whether or not an IP (SRF) is attached, and available at the SSP. This parameter is applicable in the physical scenarios corresponding to assist with relay or hand-off. The use of this parameter is network operator dependent.

12.92   IPSSPCapabilities

This parameter indicates which IP (SRF) resources are supported, and attached at the SSP.
It is applicable in the physical scenarios corresponding to assist with relay or hand-off. The use of this parameter is
network operator dependent.
For CAMEL it indicates which SRF resources supported within the VMSC/GMSC the SSF resides in are attached and
available.

12.93   ISDNAccessRelatedInformation

This parameter contains (possibly multiple) information elements as per ITU-T Recommendation Q.931. It carries the
same information as the protocol element ISUP Access Transport parameter in ITU-T Recommendation Q.763. Refer to
Access Transport signalling information in ITU-T Recommendation Q.762 for the actual definition of this parameter.

12.94   LegID

This parameter contains an identifier for a leg (communication path to e.g. a call party/network entity) in the call to
which the operation shall apply. Refer to ITU-T Recommendation Q.1290 for the actual definition of 'leg'.

OPTIONAL for LegID denotes network operator specific use with a choice of unilateral ID assignment or bilateral ID
assignment.

OPTIONAL for LegID also denotes the following:

When only one party exists in the two-party call, this parameter is not needed (as no ambiguity exists); when more than
one party exists in the call, one of the following alternatives applies:

1) LegID is present and indicates explicit which party is concerned.

2) LegID is not present and a default value is assumed.

It comprises the following alternatives:

-   sendingSideID
    This subparameter is used where legID is sent from the SCF to the SSF; or

-   receivingSideID
    This subparameter is used where SCF receives legID from the SSF.

12.95   LegIDToMove

For a definition of this sub-parameter, refer to the 'LegID' clause. It is used to contain an identifier for the leg that shall
be moved.

12.96   Legs

This parameter indicates the leg in the source CSA and the equivalent leg in the target CSA. It is used to change the
logical identifier for a leg (legID) as a call segment is being moved from a source CSA into a new (target) CSA.
It comprises the following sub-parameters.

-   sourceLeg
    For a definition of this sub-parameter, refer to the 'LegID' clause. It is used to specify a source leg identifier,
    e.g. for the leg to be moved;

-   newLeg
    For a definition of this sub-parameter, refer to the 'LegID' clause
    It is used to specify a new (target) leg identifier, e.g. for the leg that has been moved.

12.97   LegorCSID

This parameter indicates how call processing is to be resumed (leg or CS). It comprises the following alternative sub-parameters:

- legID
  For a definition of this sub-parameter, refer to the 'LegID' clause
  This alternative indicates the party (leg) for which call processing is to be resumed; or

- callSegmentID
  For a definition of this sub-parameter, refer to the 'CallSegmentID' clause
  This alternative indicates the call segment for which call processing is to be resumed.

12.98   LegToBeCreated

For a definition of this parameter, refer to the 'LegID' clause.
It is used to indicate the LegID to be assigned to a newly created party.

12.99   LegToBeReleased

For a definition of this parameter, refer to the 'LegID' clause.
It is used to indicate the party in the call to be released.

12.100 LegToBeSplit

For a definition of this parameter, refer to the 'LegID' clause.
It is used to indicate the party (leg) in the call to be split from its Call Segment.

12.101 LocationInformation

This parameter indicates the whereabouts of the MS, and the age of the information defining the whereabouts.
Refer to TS 129 002 for the definition of this parameter.

12.102 LocationNumber

This parameter is used to convey the geographical area address for mobility services, see Q762. It is used when "callingPartyNumber" does not contain any information about the geographical location of the calling party (e.g. origin dependent routing when the calling party is a mobile subscriber).

12.103 MaximumNumberOfCounters

This parameter provides the number of counters to be allocated as well as the number of destinations included in the service filtering, i.e. "maximumNumberOfCounters" subsequent destination addresses beginning with the destination address provided in "filteringCriteria" are used for service filtering. One counter is assigned to each of these destination addresses. The number of counters may only be > 1 if the "filteringCriteria" are of the type "addressAndService".

12.104 MiscCallInfo

This parameter indicates detection point related information (DP type and trigger assignment). It comprises the following sub-parameters:

- messageType
  This sub-parameter indicates whether the message is a request, i.e. resulting from a "RequestReportBCSMEvent" with monitorMode = interrupted, or a notification, i.e. resulting from a "RequestReportBCSMEvent" with "monitorMode" = "notifyAndContinue".

12.105 MonitorDuration

This parameter indicates how long the SSF should monitor the specified event. The unit is second.

12.106 MonitorMode

This parameter indicates the monitored mode for an event (i.e. a BCSM event, a charging event …).

12.107 MscAddress

This parameter gives the mscId assigned to the MSC. Refer to TS 129 002 for the definition of this parameter.

12.108 NA-OliInfo

This parameter contains originating line information, which identifies the charged party number type to the carrier.

12.109 NewCallSegment

For a definition of this parameter, refer to the 'CallSegmentID' clause.
It is used to indicate the CS ID to be assigned to the newly created Call Segment. When not provided, a default CSID of 1 is assumed ('new CSA' case). When used within the context of an existing relationship, this parameter shall be provided by the SCF.

12.110 NewCallSegmentAssociation

This parameter specifies the new CSAID.

12.111 OCSIApplicable

This parameter indicates that the Originating CAMEL Subscription Information, if present, shall be applied on the outgoing call leg created with the Connect operation. For the use of this parameter see TS 123 078.

12.112 OneTriggerResult

This parameter indicates the result of one trigger management activity. It comprises the following sub-parameters:

- actionPerformed
  For a definition of this sub-parameter, refer to the 'ActionPerformed' clause;

- extensions
  This sub-parameter is defined in EN 301 931-1.

12.113 OriginalCalledPartyID

This parameter identifies the original called party when a call has been redirected.
Refer to ITU-T Recommendation Q.762 Original Called Number for the actual definition of this parameter.
This parameter carries the dialled digits if the call is forwarded by a SCF.

12.114 PartyToCharge

This parameter indicates the leg where the charging information is to be sent.

12.115 PartyToConnect

This parameter indicates how the connection to a SRF resource for a subsequent user interaction shall be set up, that is either to a single party (leg) or all parties (call segment) in the call segment. It comprises the following alternatives:

- legID:
  For a definition of this sub-parameter, refer to the 'LegID' clause.
  This parameter indicates to which party in the call the subsequent user interaction shall apply while maintaining the speech connection between that leg and any other legs connected to the same CS; or

- callSegmentID:
  For a definition of this sub-parameter, refer to the 'CallSegmentID' clause.
  It is used to indicate to which call segment the subsequent user interaction shall apply, i.e. to all parties connected to the call segment.

12.116 PartyToDisconnect

This parameter indicates how the connection to a SRF resource shall be released, that is either to disconnect the connection for a single party (leg) or all parties (call segment) in the call segment. It comprises the following alternatives:

- LegID
  For a definition of this sub-parameter, refer to the 'LegID' clause.
  This sub-parameter indicates to which party in the call the resource is currently connected; or

- CallSegmentID
  For a definition of this sub-parameter, refer to the 'CallSegmentID' clause.
  This sub-parameter indicates to which call segment the resource is currently connected.

12.117 ProfileAndDP

This parameter identifies the line/subscriber profile linked to the created trigger DP
It identifies the service information related to the BCSM. It comprises the following sub-parameters:

- triggerID
  For a definition of this sub-parameter refer to 'EventTypeBCSM' clause.
  It identifies the DP where the TDP is applied;

- profile
  For a definition of this sub-parameter, refer to the 'Profile' clause;

- extensions
  Refer to EN 301 931-1 for a definition of this sub-parameter.

12.118 Profile

This parameter identifies the profile (line/group of line) that is to be managed. It comprises the following alternatives:

- access
  This alternative identifies a subscriber access. It provides several addressing schemes to identify the line/subscriber profile linked to a TDP. It contains the number (e.g. call party directory number) needed to identify the subscriber profile; or

- group
  This alternative identifies a facility group. It comprises the following fields:

  - trunkGroupID
    This field identifies a trunk group;

  - privateFacility
    This field indicates the particular group of private facilities to route a call;

  - huntGroup
    This field contains a link to a multi-line hunt group. Its content is network operator specific;

  - routeIndex
    This field contains a link to a specific trunk routing group. Its content is network operator specific.

12.119 RedirectingPartyID

This parameter contains the directory number of the last redirecting party. The information is sent in the forward direction when a call is diverted, indicating the number from which the call was diverted. Refer to ITU-T Recommendation Q.762 'Redirecting Number' for the actual definition of this parameter.

12.120 RedirectionInformation

This parameter contains information about call redirection (e.g. redirecting reason, redirection counter). Refer to ITU-T Recommendation Q.762 'Redirection Information' for the actual definition of this parameter.

12.121 RegistratorIdentifier

This parameter provides a reference which may be used by the SCF; e.g. in case a ManageTriggerData operation needs to be invoked by the SCF on the created DP in the CCF/SSF.
It indicates the SCF reference, which is to be verified by the SSF against the administrated information, associated with the TDP.

12.122 ReleaseCause

For a definition of this sub-parameter, refer to the 'Cause' clause.

12.123 ReportCondition

This parameter specifies the report condition when the result of the requested monitor is sent to SCF.
One of the following values can be specified:

- *statusReport*
  This value means that the status change/match of the requested monitor has occurred. In this case, the "reportCondition" parameter may not necessarily be sent;

- *timerExpired*
  This value means that the monitor duration timer has expired. In this case, the "reportCondition" parameter should be sent explicitly;

- *cancelled*
  This value means that the cancellation of the requested monitor has occurred. In this case, the "reportCondition" parameter should be sent explicitly.

12.124 RequestedInformationList

This parameter specifies a list of (1 to n) requested information specific items (RequestedInformation).
For each RequestedInformation, it comprises the following sub-subparameters.

According to the requested information the SSF sends the appropriate types and values to the SCF.

- requestedInformationType
  This sub-parameter indicates the value of one of the following requested information type:
  callAttemptElapsedTime, callStopTime, callConnectedElapsedTime, calledAddress and releaseCause;

- requestedInformationValue
  This sub-parameter contains the value of the related requested information type;

- *value of callAttemptElapsedTime*
  This value indicates the duration between the end of INAP processing of operations initiating call setup
  ("Connect", "CollectInformation", "Continue"" and "ContinueWithArgument") and the received answer
  indication from the indicated or default called party side.
  For a calling party leg it shall be set to 0.
  In case of unsuccessful call setup the network event indicating the unsuccessful call setup stops the measurement
  of "callAttemptElapsedTime";

- *value of callStopTime*
  This value indicates the time stamp when the connection to the indicated or default party is released;

- *value of callConnectedElapsedTime*
  This value indicates the duration between the received answer indication from the indicated or default called
  party side and the release of that connection or party.
  For a calling party leg it indicates the duration between the sending of InitialDP and the release of that party;

- *Value of calledAddress*
  This value indicates the incoming called party address that was received by the SSF (i.e. before translation by the
  SCF) and is as available on the UNI or NNI and interpreted as per the numbering plan;

- *Value of releaseCause*
  This value indicates the release cause that applied to the indicated or default party.

12.125 RequestedInformationTypeList

This parameter specifies a list of (1 to n) of RequestedInformationType specific items of information which is requested. For each RequestedInformationType item, it comprises the following sub-subparameters:

Any set of these values can be requested.

The list may contain:

- callAttemptElapsedTime
  This parameter indicates the request for callAttemtElapsedTime. For a called party it indicates the duration between the end of INAP processing of operations initiating call setup ("Connect", "CollectInformation", "Continue" and "ContinueWithArgument") and the received answer indication from the indicated or default called party side. For a calling party leg it is not applicable, i.e. the value will be reported as 0.
  In case of unsuccessful call setup the network event indicating the unsuccessful call setup stops the measurement of "callAttemptElapsedTime";

- callStopTime
  This parameter indicates the request for a time stamp when the connection to the indicated or default party is released;

- callConnectedElapsedTime
  This parameter indicates the request for callConnectedElapsedTime. For a called party it indicates the duration between the received answer indication from the indicated or default called party side and the release of that connection or party. For a calling party leg it indicates the duration between the sending of InitialDP and the release of that party;

- calledAddress
  This parameter indicates the request for the incoming called party address that was received by the SSF (i.e. before translation by the SCF) and is as available on the UNI or NNI and interpreted as per the numbering plan;

- releaseCause
  For a definition of this sub-parameter, refer to the 'Cause' clause.
  This sub-parameter indicates the request for the release cause that applied to the indicated or default party.

12.126 ResourceAddress

This parameter identifies the physical location of the SRF. It comprises the following alternatives:

- iPRoutingAddress
  This parameter indicates the routing address to set up a connection towards the SRF; or

- legID
  For a definition of this sub-parameter, refer to the 'LegID' clause.
  It is used to indicate to which party in the call the subsequent user interaction shall apply while maintaining the speech connection between that leg and any other legs connected to the same CS; or

- callSegmentID
  For a definition of this sub-parameter, refer to the 'CallSegmentID' clause.
  It is used to indicate to which call segment the subsequent user interaction shall apply, i.e. to all parties connected to the call segment; or

- iPAddressAndLegID
  This parameter indicates that both legID and iPRoutingAddress shall be used.
  It comprises the following fields:

  - iPRoutingAddress;

  - legID; or

- none
  This parameter indicates that the call party is to be connected to a predefined SRF in the single CS; or

- iPAddressAndCallSegment
  This parameter indicates that both Call Segment ID and iPRoutingAddress shall be used.
  It comprises the following fields:

  - iPRoutingAddress;

  - callSegmentID.

12.127 ResourceID

This parameter specifies the particular resource. It indicates that physical termination resource which is requested by the SCF to be monitored by the SSF. It comprises the following alternatives:

- lineID
  This alternative specifies a destination number (ID for a line); or

- facilityGroupID
  This alternative specifies a facility group ID (ID for a hunt group); or

- facilityGroupMemberID
  This alternative specifies a facility group member (ID for a hunt group member); or

- trunkGroupID
  This alternative specifies a trunk group ID (ID for a trunk group).

12.128 ResourceStatus

This parameter indicates the status of a termination resource. It contains the detected status (e.g. busy/idle) of a physical termination resource, which was requested to be monitored.

12.129 ResponseCondition

This parameter is used to identify the reason why the ServiceFilteringResponse is sent.

When responseCondition is intermediate response it identifies that service filtering is active, a call is received and the interval timer is expired, or that service filtering is active and the threshold value "numberOfCalls" is reached.
If the responseCondition is last Response it identifies either that the duration timer is expired and service filtering is stopped or that the stop time is met and service filtering is stopped.

12.130 RouteList

This parameter represents the list of routes used in order to route the call. It is used to select the outgoing trunk group used for routing the call. A sequence of routes is provided to allow flexible routing for applications such as VPN without increasing the number of queries required for such applications. The network operators can specify that this parameter should be used if their particular network has the information available.

12.131 RouteingNumber

This parameter specifies the number used by the network to route the call.
Refer to ITU-T Recommendation Q.762 'Network Routeing Number' for the actual definition of this parameter.

12.132 ScfID

The means of identification of an SCF. The scfID is used in the context of a hand-off/assist procedure and only if the scfID is not embedded in the 'destinationRoutingAddress'. It may also be used alone (e.g. for mobile applications) to perform a "hand-off" from an initiating SSF (e.g. in a visiting mobile domain) to a requesting SSF (e.g. in home mobile domain). The scfID is used to provide the INAP address of the SCF to establish a connection between the requesting SSF and the specified SCF. The scfID is to convey the necessary SCF address information (e.g. Global Title) in the network to the requesting SSF. See ITU-T Recommendation Q.713 "calling party address" parameter. The network operator has to decide about the actual mapping of this parameter on the used signalling system.
This parameter may also indicate the address of the SCF, which initiated the call gapping.
When ScfID is used in an operation, which may cross an internetwork boundary, its encoding must be understood in both networks; this requires bilateral agreement on the encoding.

12.133 SCIBillingChargingCharacteristics

This parameter indicates billing and/or charging characteristics. Its content is network operator specific. Depending on the applied charging scenario different information elements can be included (refer to clause 16 charging scenarios):
It comprises the following fields:

- charge level;

- chargePulses;

- chargeMessages.

12.134 SDSSinformation

This parameter contains the "application" layer SDSS protocol. It is used by the SCF to interact with the analogue user by means of the SDSS (Server Display and Script Services) protocol on the analogue interface.

12.135 SendCalculationToSCPIndication

This parameter indicates that "ApplyChargingReport" operations (at least one at the end of the release of the connection) are expected from the SSF. This parameter, *when provided,* is always set to TRUE. It is ignored if CS-2 or CS-3 is used.

12.136 ServiceInteractionIndicators

This parameter contains indicators sent from the SCF to the SSF for control of the network-based services at the originating and the destination exchange. Its content is network signalling/operator specific.

NOTE: This parameter is kept in the present document for backward compatibility to IN CS-1R, for the present document see parameter ServiceInteractionIndicatorsTwo.

12.137 ServiceInteractionIndicatorsTwo

This parameter contains a set of indicators which are exchanged between SSP and SCP to resolve interactions between IN based services and network based services, respectively between different IN based services. It comprises the following sub-parameters:

- forwardServiceInteractionInd;
  It is applicable to operations IDP, CON, ICA, CWA, SF.
  This field contains indicators applicable to be sent in forward direction to activate or deactivate suppression of switch based services at the destination local exchange. Refer to ITU-T Recommendation Q.1601.
  It comprises the following fields:

  - conferenceTreatmentIndicator
    This field indicates if a network switch based conference request is to be accepted or rejected.
    The network default is accept conference request;

  - callDiversionTreatmentIndicator
    This field indicates if a network switch based call diversion service request is to be allowed or not allowed,
    The network default is Call Diversion allowed;

  - callOfferingTreatmentIndicator
    This field indicates if call offering is "allowed"," not allowed" or "no impact by IN"
    - the value 'no impact by IN," has only local significance in SSF as a request to SSF not to modify the value of the call offering treatment indicator conveyed in signalling. The network default is Call Offering not allowed;

  - callingPartyRestrictionIndicator
    This field allows the SCF to control the presentation restriction of the calling party number. It indicates the setting of the presentation indicator to "presentation restricted". The default is "no IN impact";

  - callWaitingTreatmentIndicator
    This field indicates if a network switch based call waiting service request is to be allowed or not allowed, The network default is Call Waiting allowed;

  - holdTreatmentIndicator
    It is applicable to operations IDP, CON, CWA, SF.
    This field indicates if a network switch based call hold service request is to be accepted or rejected, The network default is accept Call Hold request;

  - ectTreatmentIndicator
    It is applicable to operations IDP, CON, CWA, SF.
    This field indicates whether the call leg can become part or not of an ECT call, i.e. if the ECT service request is to be accepted or rejected.
    The network default is accept ETC request.

- backwardServiceInteractionInd
  It is applicable to operations IDP, CON, CTR, ETC, CWA, SF.
  This field contains indicators applicable to be sent in backward direction to activate or deactivate suppression of switch based services at the originating local exchange. Refer to ITU-T Recommendation Q.1601.
  It comprises the following fields:

  - conferenceTreatmentIndicator
    This field indicates if a network switch based conference request is to be accepted or rejected.
    The network default is accept conference request;

  - callCompletionTreatmentIndicator
    This field indicates if a network switch based call completion service request is to be accepted or rejected,
    The network default is accept call completion service request;

  - holdTreatmentIndicator
    It is applicable to operations IDP, CON, CWA, SF.
    This field indicates if a network switch based call hold service request is to be accepted or rejected, The network default is accept Call Hold request;

- ectTreatmentIndicator
  It is applicable to operations IDP, CON, CWA, SF.
  This field indicates whether the call leg can become part or not of an ECT call, i.e. if the ECT service request is to be accepted or rejected.
  The network default is accept ECT request;

- bothwayThroughConnectionInd
  It is applicable to operations CTR, ETC.
  This field allows the SCF to control bothway through connection in backward direction for user interaction, i.e. to indicate bothway through connection ' required' or 'not required'. Default is 'required'. Refer to ITU-T Recommendation Q.1601;

- suspendTimer
  It is applicable to operations CON, ICA CWA, SF.
  This field allows the SCF to control the suspend timer to control the release of a called party (non-ISDN subscriber). A reduced time supervision or an immediate release for the suspend timer as compared to the normal networked timer (T6) is possible to specify.
  The suspend timer is to be started by SSF upon receipt of a suspend message generated by the network in response to a clear back indication from an interworking node or an on-hook condition from an analogue called party. Refer to ITU-T Recommendation Q.1601;

- connectedNumberTreatmentInd
  It is applicable to operations CON, CTR, ETC, CWA, SF.
  This field allows the SCF to control the number to be presented as 'connected' number to calling party.
  The following alternatives are possible:

  - noINImpact; or

  - presentationRestricted; or

  - presentCalledINNumber; or

  - presentCalledINNumberRestricted;

- suppressCallDiversionNotification
  It is applicable to CON, ICA, CWA, SF
  Refer to ITU-T Recommendation Q.1601;

- suppressCallTransferNotification
  It is applicable to CON, ICA, CWA, SF
  Refer to ITU-T Recommendation Q.1601;

- allowCdINNoPresentationInd
  It is applicable to CON, ICA, CWA, SF
  It indicates whether the Address Presentation restricted indicator of the ISUP "called IN number" shall be set to presentation allowed (TRUE) or presentation restricted (FALSE). Refer to ITU-T Recommendation Q.1601;

- userDialogueDurationInd
  It is applicable to operations CTR, ETC.
  It is applicable when interaction with the user is required during call set-up. The interaction TRUE means the user interaction may last longer than 90 s. Otherwise the indicator should be set to FALSE. Used for delaying ISUP T9 timer (answer timer). The default is TRUE. Refer to ITU-T Recommendation Q.1601;

- overrideLineRestrictions
  It is only applicable to operations (e.g. Connect) which lead to a transition to a PIC before the AuthorizeCallSetup PIC. When set to TRUE, this parameter indicates that some facility restrictions should not be checked when the authority to place a call is verified in the Authorize_Call_Setup PIC.
  Which restrictions are actually overridden is network specific.
  The default is not to override line restrictions (i.e. FALSE);

- suppressVPNAPP
  It is applicable to CWA, CON, ICA, SF.
  This field indicates whether to allow or stop (suppress) the forward transmission of the VPN PSS1 capability.
  When set to TRUE, the exchange, on receipt of this parameter, will not transmit for this call
  any ISUP Application transport parameter with Application Context Identifier set to « PSS1 ASE (VPN) ».
  This indicator is populated by the SCF, where the SCF and SSF in conjunction have provided the outgoing
  gateway PINX functionality as required by PSS1. The default is "allow" (i.e. FALSE);

- calledINNumberOverriding
  It is applicable to CON and CWA, SF
  It indicates whether the generation/override of the ISUP "called IN number" is allowed (TRUE) or not allowed
  (FALSE) If set to FALSE, the ISUP shall not generate a "called IN number" or override an already existing
  "called IN number".
  If absent, the default will be "generation/overriding allowed" (TRUE);

- nonCUGCall
  It is applicable for Connect and CWA.
  It indicates that no parameters for CUG shall be used for the call (i.e. the call shall be a non-CUG call).
  When this indicator is present, the following applies:

  - continue with modified CUG information (when one or more of either CUG Interlock Code and Outgoing
    Access Indicator are present); or

  - continue with original CUG information (when neither CUG Interlock Code nor Outgoing Access Indicator
    is present).

12.138 ServiceKey

This parameter indicates the service key value associated with the trigger (TDP) in the CCF/SSF. It identifies for the
SCF unambiguously the requested IN service. It is used to address the correct application/SLP within the SCF (not for
SCP addressing).

12.139 SeveralTriggerResult

This parameter contains a list of (1 to n) trigger result. For each trigger result, it comprises the following
sub-parameters:

- triggerResult
  This sub-parameter indicates the result of the operation for an individual trigger (activated, deactivated, TDP
  status). It comprises the following fields:

  - tDPIdentifer
    For a definition of this field, refer to the 'TDPIdentifier' clause;

  - actionPerformed
    For a definition of this field, refer to the 'ActionPerformed' clause;

  - dPName
    For a definition of this sub-parameter, refer to the 'DPName' clause.

12.140 SourceCallSegment

For a definition of this sub-parameter, refer to the 'CallSegmentID' clause.
This parameter indicates the source CS for the operation; e.g. the CS that shall be merged with another (target) CS and
after the merge will be deleted.

12.141 StartTime

This parameter defines the date and time when service filtering is started.

12.142 SubscriberState

The state of the mobile subscriber for which the service is invoked. The possible states are busy, idle and not reachable. For encoding see TS 129 002.

12.143 SuppressionOfAnnouncement

This parameter indicates that announcements and tones which are played in the exchange at non-successful call set-up attempts shall be suppressed.

12.144 TargetCallSegment

This parameter indicates the target CS for the operation; e.g. the CS that shall be merged with another (source) CS.

12.145 TargetCallSegmentAssociation

This parameter indicates the target Call Segment Association, e.g. the CSA into which a CS from another (source) CSA is to be moved.

12.146 TariffMessage

This parameter includes the contents of the charging tariff message. It may include tariff -, add-on charge -, acknowledgement -, start - or stop information. The information is contained in the related ChargingTariffInformation -, AddOnChargingInformation -, ChargingAcknowledgementInformation, StartCharging or StopCharging sub-parameters. The details of interworking for this parameter are defined within ES 201 296 version 2.

> NOTE: The sub-parameter 'destination identification' within the ChargingTariffInformation, the AddOnChargingInformation, the StartCharging or StopCharging is not relevant.

12.147 TDPIdentifier

This parameter identifies one or more triggers to be managed, for example in order to activate, deactivate or remove a trigger. It comprises the following alternatives:

- oneTrigger
  This alternative contains the local identifier assigned by the SSF to the created trigger (i.e. the service trigger Identifier); or

- triggers
  This alternative contains a list of (1 to n) Trigger.
  For each Trigger it comprises the following fields:

  - tDPIdentifier
    This parameter contains the local identifier assigned by the SSF to the created trigger (i.e. the service trigger Identifier);

  - dPName
    For a definition of this field, refer to the 'DPName' clause.

12.148 TerminalType

This parameter indicates the type of terminal (e.g. DTMF phone, ISDN terminal).
For example in order to allow that the SCF can specify to the SRF, the appropriate type of capability (voice recognition, DTMF, display capability, etc.).

> NOTE: Since present signalling systems do not convey terminal type, this parameter applies only at originating or terminating local exchanges.

12.149 TimeAndTimezone

This parameter contains the time that the SSF was triggered, and the time zone that the invoking SSF resides in.

12.150 TimerID

This parameter has a default value identifying the T_{SSF} timer.

12.151 TimerValue

This parameter specifies the value to which the T_{SSF} is to be set.

12.152 TimeToRelease

If this parameter is present a timed disconnect is requested. It indicates the time the SSF waits before releasing the call (i.e. all call segments). The timer is set upon receipt of the ReleaseCall operation.

12.153 TriggerData

This parameter provides service specific static information associated with the created trigger including trigger criteria. Its content is network operator specific.

12.154 TriggerDPType

This parameter indicates the DP trigger type TDP-R.

12.155 TriggerStatus

This parameter provides the result in case the procedure is successfully executed.
The trigger status result may indicate if a trigger is created, already exists, is deleted or unknown.

12.156 TriggerDataIdentifier

This parameter identifies the service information related to the BCSM: the DP where the TDP is applied and the corresponding subscriber profile (line/group of line) that is to be managed.

12.157 USIInformation

This parameter conveys information from the User dedicated to e.g. the Service Logic. It is transparent at the SSF level.

12.158 USIMonitorMode

This parameter indicates if the USI information is to be monitored. When the "USImonitorMode" is "monitoringActive" the received USI Information (provided match the requested USIServiceIndicator value) is reported as a notification. When the "USImonitorMode" is "monitoringInactive", the USI Information is not (or no more) reported.

12.159 USIServiceIndicator

This parameter indicates the Service Logic, i.e. the user requesting the Monitoring of an USI information element or the sending of an USI information element. It is used as monitor criteria at the SSF level. It also provides the correlation with the SCF for the RequestReportUTSI operation.

12.160 VPNIndicator

This parameter indicates that the call is a VPN call that has potential PSS1 capabilities. This indication is based on information received in the originating call setup, i.e. the presence of the VPN indicator information element in the DSS1 SETUP message or the ACI coded « PSS1 ASE(VPN) » in the ISUP IAM message.

13 Errors

This clause defines the error procedures for the SSF – SCF interface. Error descriptions are provided in EN 301 931-1 and the following clauses provide operation related error procedures and when relevant, error procedures related to error conditions which are not directly related to the failure of an INAP operation.

13.1 Operation related error procedures

The following clauses defines the generic error handling for the operation related error procedures on the SSF – SCF interface. The errors are defined as operation errors in the ASN.1 operations related description. The TC services which are used for reporting operation errors are described in clause 15.

Table 38: Abbreviations for Operations on SSF-SCF interface

Operation abbreviation	Operations SCF -> SSF Operation Name	Operation abbreviation	Operations SSF -> SCF: Operation Name
AC ASF	ApplyChargingActivateServiceFilteringCancel	ACR	ApplyChargingReport
Ca	CancelStatusReportRequest	ARI	AssistRequestInstructions
CaSRR CCSA	CreateCallSegmentAssociation CollectInformation	IDP	InitialDP
CI	CallInformationRequest		
CIR	Connect		
Co CRTD	CreateOrRemoveTriggerData ConnectToResource		
CTR	ContinueWithArgument		
CWA	DisconnectForwardConnection		
DFC	DisconnectForwardConnectionWithArgument		
DFCWA	DisconnectLeg		
DL	EstablishTemporaryConnection		
ETC	FurnishChargingInformation		
FCI	InitiateCallAttempt		
ICA MeCS MoCS ML MTD RCSR RESCR	MergeCallSegments MoveCallSegments MoveLeg ManageTriggerData RequestCurrentStatusReport RequestEveryStatusChangeReport		
RFSMR	RequestFirstStatusMatchReport		
RNCE	RequestNotificationChargingEvent		
RRB	RequestReportBCSMEvent		
RRUTSI	RequestReportUTSI		
RT	ResetTimer SendChargingInformation SetServiceProfile		
SCI SSP SSTUI SF	SendSTUI SelectFacility		

All errors, which can be detected by the ASN.1 encoder, already may be detected during the decoding of the TCAP message and indicated by the TC error indication "MistypedParameter" in the TC-U-Reject.

The following tables provides the list of operations applicable for the SSF-SCF interface
In the tables listing the operations, which may return each of the errors used on the SSF – SCF interface, an "X" entry is used to indicate when an error is applicable for a listed operation.

Table 39: Available Errors for each SSF - SCF Operation

Errors	AC	ASF	Ca	CaSRR	CCSA	CI	CIR	Co	CRTD	CTR	CWA	DFC	DFCWA	DL	ETC	FCI	HCIN	ICA	MeCS	ML	MoCS	MTD	RCSR	Re	RESCR	RFSMR	RNCE	RRB
ETCFailed															X													
CancelFailed				X																								
MissingCustomerRecord									X																			
MissingParameter	X	X	X	X	X	X	X	X	X	X	X		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
NameError																		X										
ParameterOutOfRange	X	X				X	X	X	X		X				X			X	X	X	X	X	X	X	X	X	X	X
RequestedInfoError							X								X		X	X	X	X					X			
SystemFailure	X	X			X	X	X	X	X	X	X	X	X	X	X		X	X	X	X	X	X	X	X	X	X	X	X
TaskRefused	X	X		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
UnexpectedComponentSequence	X	X			X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
UnexpectedDataValue	X				X	X	X	X	X	X	X		X	X	X	X			X	X	X	X			X	X	X	X
UnexpectedParameter	X	X			X	X	X	X	X	X	X		X	X				X			X	X			X	X	X	X
UnknownLegID	X						X			X			X	X														
UnknownRessource																								X	X	X		

Table 39a: Available Errors for each SSF - SCF Operation

Errors	SCF -> SSF										SSF -> SCF		
	RRFE	RRUTSI	RT	SCI	SSF	SFI	SL	SR	SSP	SSTUI	ACR	ARI	IDP
ETCFailed													
MissingCustomerRecord												X	X
MissingParameter	X	X	X	X	X	X	X	X	X	X	X	X	X
ParameterOutOfRange			X	X	X			X			X		X
SystemFailure	X	X	X	X	X		X	X	X	X	X	X	X
TaskRefused		X	X	X			X	X			X	X	X
UnexpectedComponent-Sequence	X	X	X	X	X	X	X	X		X	X	X	X
UnexpectedDataValue	X	X	X	X	X	X	X	X	X	X	X	X	X
UnexpectedParameter	X	X	X	X	X	X	X	X	X	X	X	X	X
UnknownLegID	X	X	X	X		X	X		X	X			

13.1.1 CancelFailed

The Error "CancelFailed" is defined in the EN 301 931-1.

13.1.1.1 Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

13.1.1.1.1 Procedures at invoking entity (SCF)

A) SCF sends the operation to SSF
Precondition: refer to the relevant "Operation procedures - SCF - precondition" clause.
Postcondition: refer to the relevant "Operation procedures - SCF - postcondition" clause.

B) Receiving CancelFailed Error
Precondition: refer to the relevant "Operation procedures - SCF - postcondition" clause.
Postcondition: SCSM state: remains in the same state.

After receiving a CancelFailed Error the Service Logic may continue. The CancelFailed Error can therefore be received in any state. The treatment is Service Logic dependent.

13.1.1.1.2   Procedures at responding entity (SSF)

A) Receiving the operation
Precondition: refer to the relevant "Operation procedures SRF - precondition" clause.
Postcondition: refer to the relevant "Operation procedures SRF - postcondition" clause.

The monitoring of the indicated resource is terminated if it is presently active. If the monitoring of the indicated resource is already terminated this causes a failure ("CancelFailed") to the CancelStatusReportRequest.

B) Sending CancelFailed Error
Precondition: refer to the relevant "Operation procedures –SSF- postcondition" clause.
Postcondition: remain in the same state.

13.1.2   ETCFailed

The Error "CancelFailed" is defined in the EN 301 931-1.

13.1.2.1   Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

13.1.2.1.1   Procedures at invoking entity (SCF)

A) SCF sends the Operation to SSF
Precondition: refer to the relevant "Operation procedures - SCF - precondition" clause.
Postcondition: refer to the relevant "Operation procedures - SCF - postcondition" clause.

B) Receiving ETCFailed Error
Precondition: refer to the relevant "Operation procedures - SCF - postcondition" clause.
Postcondition: SCSM state: remains in the same state.

13.1.2.1.2   Procedures at responding entity (SSF)

A) Receiving Operation
Precondition: refer to the relevant "Operation procedures SSF - precondition" clause.
Postcondition: refer to the relevant "Operation procedures SSF - postcondition" clause.

Until the connection setup has been accepted by the assisting SSF/SRF (i.e. receipt of CallProgreessInd(bptyAlerted)/CallProgressReqInd(bptyAlerted) or SetupConf/SetupRespConf (direct answer) primitive), all received failure indications from the network on the ETC establishment shall be reported to the SCF as ETC error ETCFailed (e.g. busy, congestion).
Note that the operation timer for ETC shall be longer then the maximum allowed time for the signalling procedures to accept the connection.

B) Sending ETCFailed Error
Precondition: refer to the relevant "Operation procedures –SSF- postcondition" clause.
Postcondition: remain in the same state.

13.1.3   MissingCustomerRecord

The Error "MissingCustomerRecord" is defined in EN 301 931-1.

13.1.3.1   Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

13.1.3.1.1 Procedures at invoking entity (SCF)

A) SCF sends the Operation to SSF
   Precondition: refer to the relevant "Operation procedures - SCF - precondition" clause.
   Postcondition: refer to the relevant "Operation procedures - SCF - postcondition" clause.

B) Receiving MissingCustomerRecord Error
   Precondition: refer to the relevant "Operation procedures - SCF - postcondition" clause.
   Postcondition: SCSM state: remains in the same state.

   The Service Logic and maintenance functions are informed. Further treatment of the call is dependent on Service Logic.

13.1.3.1.2 Procedures at responding entity (SSF)

A) Receiving Operation
   Precondition: refer to the relevant "Operation procedures SSF - precondition" clause.
   Postcondition: refer to the relevant "Operation procedures SSF - postcondition" clause.

B) Sending MisssingCustomerRecord Error
   Precondition: refer to the relevant "Operation procedures - SSF- postcondition" clause.
   Postcondition: remain in the same state.

13.1.3.2 Operations SSF -> SCF

This clause describes the procedure when the error for an operation invoked from the SSF occurs in the SCF.

Relevant operations are described in table 39.

13.1.3.2.1 Procedures at invoking entity (SSF)

A) SSF sends the Operation to SCF
   Precondition: refer to the relevant "Operation procedures - SSF - precondition" clause.
   Postcondition: refer to the relevant "Operation procedures - SSF - postcondition" clause.

B) Receiving MissingCustomerRecord Error
   Precondition: refer to the relevant "Operation procedures - SSF - postcondition" clause.
   Postcondition: SSF FSM state 'Idle'.

The CCF routes the call if necessary (e.g. default routing to a terminating announcement).

13.1.3.2.2 Procedures at responding entity (SCF)

A) Receiving Operation
   Precondition: refer to the relevant "Operation procedures SCF - precondition" clause.
   Postcondition: refer to the relevant "Operation procedures SCF - postcondition" clause.

B) Sending MisssingCustomerRecord Error
   Precondition: refer to the relevant "Operation procedures - SCF - postcondition" clause.
   Postcondition: SCSM state: remains in the same state.

13.1.4 MissingParameter

The Error "MissingParameter" is defined in EN 301 931-1.

13.1.4.1 Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

13.1.4.1.1   Procedures at invoking entity (SCF)

A) Sending Operation
   Precondition: refer to the relevant "Operation procedures - SCF - precondition" clause.
   Postcondition: refer to the relevant "Operation procedures - SCF - postcondition" clause.

B) SCF receives Error "MissingParameter"
   Precondition: refer to the relevant "Operation procedures - SCF - postcondition" clause.
   Postcondition: SCSM state: remain in the same state.

The Service Logic and maintenance functions are informed. Further treatment of the call is dependent on Service Logic.

13.1.4.1.2   Procedures at responding entity (SSF)

Precondition: refer to the relevant "Operation procedures SSF - precondition" clause.
Postcondition: remain in the same state.

The SSF detects that a required parameter is not present in the Operation argument and makes a transition to the initial state (i.e. before receiving the erroneous operation).

The Error parameter MissingParameter is used to inform the SCF of this situation. The SCF should take the appropriate actions to treat this error.

13.1.4.2   Operations SSF -> SCF

This clause describes the procedure when the error for an operation invoked from the SSF occurs in the SCF.

Relevant operations are described in table 39.

13.1.4.2.1   Procedures at invoking entity (SSF)

A) Sending Operation
   Precondition: refer to the relevant "Operation procedures SRF - precondition" clause.
   Postcondition: refer to the relevant "Operation procedures SRF - postcondition" clause.

B) SSF receives Error "MissingParameter"
   Precondition: refer to the relevant "Operation procedures - SRF - postcondition" clause.
   Postcondition: SSF FSM state 'Idle'.

After receiving this Error, the SSF FSM returns to the state Idle. The CCF routes the call if necessary (default routing to a terminating announcement). If the call is already established (i.e. mid-call trigger or ApplyChargingReport), the CCF may maintain the call or disconnect it. The choice between these two options is network operator specific. In case of an assisting SSF, the temporary connection is released by the assisting SSF.

13.1.4.2.2   Procedures at responding entity (SCF)

Precondition: refer to the relevant "Operation procedures SCF - precondition" clause.
Postcondition: SCSM state: 'Idle'.

The SCSM detects the erroneous situation. The Error parameter is used to inform the SSF of this situation. The Service Logic and maintenance functions are informed.

13.1.5   ParameterOutOfRange

The Error "ParameterOutOfRange" is defined in EN 301 931-1.

13.1.5.1   Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.5.2   Operations SSF -> SCF

This clause describes the procedure when the error for an operation invoked from the SSF occurs in the SCF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.6   RequestedInfoError

The Error "RequestedInfoError" is defined in EN 301 931-1.

13.1.6.1   Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39. Refer to clause for MissingCustomerRecord error for the appropriate error procedures.

13.1.7   SystemFailure

The Error "SystemFailure" is defined in EN 301 931-1.

13.1.7.1   Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.7.2   Operations SSF -> SCF

This clause describes the procedure when the error for an operation invoked from the SSF occurs in the SCF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.8   TaskRefused

The Error "TaskRefused" is defined in EN 301 931-1.

13.1.8.1   Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.8.2 Operations SSF -> SCF

This clause describes the procedure when the error for an operation invoked from the SSF occurs in the SCF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.9 UnexpectedComponentSequence

The Error "UnexpectedComponentSequence" error is defined in EN 301 931-1.

13.1.9.1 Operations SSF -> SCF

This clause describes the procedure when the error for an operation invoked from the SSF occurs in the SCF.

Relevant operations are described in table 39.

13.1.9.1.1 Procedures at invoking entity (SSF)

On receiving the error the assisting SSF moves to Idle and the temporary connection in case of assisting SSF is released.

In case the operation AssistRequestInstructions is sent by an "initiating" SSF in the context of an existing relationship, the SCF returns the error parameter. Service Logic and maintenance are informed. On receiving the error the "initiating" SSF moves to Idle.

13.1.9.1.2 Procedures at responding entity (SCF)

In case of assisting SSF an error occurs in case an AssistRequestInstructions is sent while a relationship between SCF and assisting SSF has already been established, the SCF returns the error parameter. Service Logic and maintenance are informed.

13.1.9.2 Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

13.1.9.2.1 Procedures at invoking entity (SCF)

In the SCF the Service Logic and maintenance functions are informed and the Service Logic decides about error treatment.

13.1.9.2.2 Procedures at responding entity (SSF)

In this case the SSF detects the erroneous situation, sends the UnexpectedComponentSequence error and remains in the same state.

13.1.10 UnexpectedDataValue

The Error "UnexpectedDataValue" is defined in EN 301 931-1.

13.1.10.1 Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.10.2   Operations SSF -> SCF

This clause describes the procedure when the error for an operation invoked from the SSF occurs in the SCF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.11 UnexpectedParameter

The Error "UnexpectedParameter" is defined in EN 301 931-1.

13.1.11.1   Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.11.2   Operations SSF -> SCF

This clause describes the procedure when the error for an operation invoked from the SSF occurs in the SCF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.12 UnknownLegID

The Error "UnknownLegID" is defined in EN 301 931-1.

13.1.12.1   Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.12.2   Operations SSF -> SCF

This clause describes the procedure when the error for an operation invoked from the SSF occurs in the SCF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.13 UnknownResource

The Error "UnknownResource" is defined in EN 301 931-1.

13.1.13.1   Operations SCF -> SSF

This clause describes the procedure when the error for an operation invoked from the SCF occurs in the SSF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.1.13.2 Operations SSF -> SCF

This clause describes the procedure when the error for an operation invoked from the SSF occurs in the SCF.

Relevant operations are described in table 39.

Refer to clause for MissingParameter error for the appropriate error procedures.

13.2 Entity related error procedures

The following clauses define the error handling for the entity related errors. Since the error situations are not originated by the reception of an operation, the invoking entity is denoted here as the entity at which the error situation is detected. The responding entity is the entity which receives the error report.

The TCAP services used for reporting errors are described in clause 15.

13.2.1 Expiration of T$_{SSF}$

13.2.1.1 General description

13.2.1.1.1 Error description

A timeout occurred in the SSF on the response from the SCF.

13.2.1.2 Procedures SSF -> SCF

13.2.1.2.1 Procedures at the invoking entity (SSF)

Timeout occurs in SSF on T$_{SSF}$.

Precondition: SSF FSM state   "Waiting for instructions" or SSF FSM "Waiting for Facility Event";

or SSF FSM state   Waiting for end of User Interaction(WFI);

or SSF FSM state h Waiting for end of User Interaction(MON);

or SSF FSM state e Waiting for end of Temporary connection(WFI);

or SSF FSM state i Waiting for end of Temporary connection(MON).

Postcondition: SSF FSM state a Idle.

The SSF aborts the dialogue and moves to the Idle state when the last CS is cleared and the CCF routes the call if necessary (e.g. default routing to a terminating announcement). The abort is reported to the maintenance functions. If not the last CS the dialogue remains and an EntityReleased operation is sent to report the error to the SCF.

13.2.1.2.2 Procedures at the responding entity (SCF)

SCF receives a dialogue abort.

Precondition: Any state.

Postcondition: SCSM state 1   Idle; if the abort is related to a SSF dialogue; or

SCSM state 2   Preparing SSF instructions; if the abort is related to an assisting SSF dialogue.

The SCF releases all allocated resources and reports the abort to the maintenance functions.

If the dialogue is aborted, the SCF releases all resources related to the dialogue, reports the abort to the maintenance functions and returns to state preparing SSF instructions.

NOTE: If SCF receives an EntityRelease the dialogue is not aborted and error treatment is service logic dependent.

14   ASN.1 definitions

14.1   Data Types

```
-- The Definition of SSF – SCF  Data Types Follows

IN-CS3-SSF-SCF-datatypes {itu-t(0) identified-organization(4) etsi(0) inDomain(1) in-network(1)
cs3(30) modules(1) in-cs3-ssf-scf-datatypes(6) version1(0)}


DEFINITIONS IMPLICIT TAGS::=

BEGIN
IMPORTS
    common-classes,
    common-datatypes,
    ssf-scf-classes,
    scf-srf-classes,
    scf-srf-datatypes,
    tc-Messages

FROM IN-CS3-object-identifiers {itu-t(0) identified-organization(4) etsi(0) inDomain(1) in-
network(1) cs3(30) modules(1) in-cs3-object-identifiers(0) version1(0)}

    COMMON-BOUNDS
FROM IN-CS3-common-classes common-classes

    TRIGGER,
    SCF-SSF-BOUNDS
FROM IN-CS3-SSF-SCF-Classes ssf-scf-classes

    SCF-SRF-BOUNDS
FROM IN-CS3-scf-srf-classes scf-srf-classes

    Extensions{},
    Integer4
FROM IN-CS3-common-datatypes common-datatypes

            InformationToSend {}

FROM IN-CS3-scf-srf-datatypes scf-srf-datatypes


    AddOnChargingInformation,
    ChargingTariffInformation,
    ChargingMessageType
FROM Tariffing-DataTypes {itu-t(0) identified-organization etsi (0) 1296 version2(3)}

    ISDN-AddressString
FROM MAP-CommonDataTypes {ITU-T(0) identified-organization(4) etsi(0) mobileDomain(0) gsm-Network(1)
modules(3) map-CommonDataTypes(18) version6(6)}
;-- The following three definitions are local short-hand notation for convenience.
B1::= COMMON-BOUNDS      -- defined in EN 301 931-1
B2::= SCF-SSF-BOUNDS        -- defined herein.
B3::= SCF-SRF-BOUNDS        -- defined in EN 301 931-3.
```

```
AChBillingChargingCharacteristics {B2: b2} ::= OCTET STRING (SIZE
        (b2.&minAChBillingChargingLength..b2.&maxAChBillingChargingLength))
```
-- The AChBillingChargingCharacteristics parameter specifies charging related information.
-- Its content is network operator specific.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this paramter (after the first tag and length
-- information) is the BER encoding of the defined ASN.1 internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.
-- CAMEL:
-- **AChBillingChargingCharacteristics** {PARAMETERS-BOUND: bound}::= OCTET STRING (SIZE
--   (bound.&minAChBillingChargingLength..bound.&maxAChBillingChargingLength))
--   (CONSTRAINED BY {
-- shall be the result of the BER-encoded value of the type --
-- **CAMEL-AchBillingChargingCharacteristics** {bound}})
-- The AChBillingChargingCharacteristics parameter specifies the charging related information
-- to be provided by the gsmSSF and the conditions on which this information has to be reported
-- back to the gsmSCF with the ApplyChargingReport operation. The value of the
-- AchBillingChargingCharacteristics of type OCTET STRING carries a value of
-- the ASN.1 data type: **CAMEL-AchBillingChargingCharacteristics**.
-- The normal encoding rules are used to encode this value.
-- The violation of the UserDefinedConstraint shall be handled as an ASN.1 syntax error.


```
AChChargingAddress  {B2: b2}    ::= CHOICE {
          legID             [2] LegID,
          srfCallSegment  [50] CallSegmentID {b2},
          bNCF            [51] LegID
}

ActionIndicator ::= ENUMERATED {
    activate        (1),
    deactivate      (2),
    retrieve        (3)
    }
```
-- indicates the action to be performed by the ManageTriggerData operation.


```
ActionOnProfile::= ENUMERATED {
    activate        (0),
    deactivate      (1)
    }
```
-- indicates the action to be performed by the SetServiceProfile operation.

```
ActionPerformed ::= ENUMERATED {
    activated       (1),
    deactivated     (2),
    alreadyActive   (3),
    alreadyInactive (4),
    isActive        (5),
    isInactive      (6),
    tDPunknown      (7)
    }
```
-- indicates the result of the operation ManageTriggerData

```
AdditionalCallingPartyNumber  {B2: b2}::= Digits {b2}
```
-- Indicates the Additional Calling Party Number. Refer to ITU-T Recommendation Q.763
-- Generic Number for encoding.

```
AlertingPattern::= OCTET STRING (SIZE(3))
```
-- Indicates a specific pattern that is used to alert a subscriber
-- (e.g. distinctive ringing, tones, etc.).
-- Only the trailing OCTET is used, the remaining OCTETS should be sent as NULL (zero)
-- The receiving side ignores the leading two OCTETS.
-- Only applies if SSF is the terminating local exchange for the subscriber.
-- Refer to the ITU-T Recommendation Q.931 Signal parameter
-- respective TS 129 002 for encoding.

```
AOCBeforeAnswer                    ::= SEQUENCE {
    aOCInitial                     [0] CAI-GSM0224,
    aOCSubsequent                  [1] AOCSubsequent   OPTIONAL,
...
    }
```
-- support for CAMEL

```
AOCSubsequent                           ::= SEQUENCE {
    cAI-GSM0224                         [0] CAI-GSM0224 ,
    tariffSwitchInterval                [1] INTEGER (1..86400)  OPTIONAL,
...
    }
--   tariffSwitchInterval is measured in 1 second units
-- support for CAMEL


ApplicationTimer::=INTEGER (0..2047)
-- Used by the SCF to set a timer in the SSF. The timer is in seconds.


AssistingSSPIPRoutingAddress  {B2: b2}::= Digits {b2}
-- Indicates the destination address of the SRF for the assist procedure.
-- Refer to ITU-T Recommendation Q.763 Generic Number for encoding.
BackwardGVNS  {B2: b2}::= OCTET STRING (SIZE(
b2.&minBackwardGVNSLength..b2.&maxBackwardGVNSLength))
-- Indicates the GVNS Backward information. Refer to ITU-T Recommendation Q.735 for encoding.


BackwardServiceInteractionInd::= SEQUENCE {
    conferenceTreatmentIndicator       [1] OCTET STRING (SIZE(1))  OPTIONAL,
    -- acceptConferenceRequest               'xxxx xx01'B
    -- rejectConferenceRequest               'xxxx xx10'B
    -- network default is accept conference request,

    callCompletionTreatmentIndicator   [2] OCTET STRING (SIZE(1))  OPTIONAL,
    -- acceptCallCompletionServiceRequest   'xxxx xx01'B,
    -- rejectCallCompletionServiceRequest   'xxxx xx10'B
    -- network default is accept call completion service request

    holdTreatmentIndicator             [3] OCTET STRING (SIZE(1))  OPTIONAL,
    -- acceptHoldRequest     'xxxx xx01'B
    -- rejectHoldRequest     'xxxx xx10'B
    -- network default is accept hold request

    ectTreatmentIndicator              [4] OCTET STRING (SIZE(1))  OPTIONAL,
    -- acceptEctRequest 'xxxx xx01'B
    -- rejectEctRequest 'xxxx xx10'B
    -- network default is accept ect request
    ...
    }


BasicGapCriteria {B2: b2} ::= CHOICE {
    calledAddressValue        [0] Digits {b2},
    gapOnService              [2] GapOnService,
    gapAllInTraffic           [3] NULL,
    calledAddressAndService   [29] SEQUENCE {
        calledAddressValue        [0] Digits {b2},
        serviceKey                [1] ServiceKey,
        ...
        },
    callingAddressAndService  [30] SEQUENCE {
        callingAddressValue       [0] Digits {b2},
        serviceKey                [1] ServiceKey,
        locationNumber            [2] LocationNumber {b2}        OPTIONAL,
    ...
        }
    }
-- Both calledAddressValue and callingAddressValue can be
-- incomplete numbers, in the sense that a limited amount of digits can be given.
-- For encoding of the digits, refer to ITU-T Recommendation Q.763 Generic Number.


BCSMEvent  {B2: b2}::= SEQUENCE {
    eventTypeBCSM     [0] EventTypeBCSM,
    monitorMode       [1] MonitorMode,
    legID             [2] LegID   OPTIONAL,
    dpSpecificCriteria [30] DpSpecificCriteria {b2}    OPTIONAL,
    ...
    }
-- Indicates the BCSM Event information for monitoring.
```

```
BearerCapability  {B2: b2}::= CHOICE {
    bearerCap         [0] OCTET STRING
        (SIZE(2..b2.&maxBearerCapabilityLength)),
    tmr               [1] OCTET STRING (SIZE(1))

    }
-- Indicates the type of bearer capability connection to the user.
-- For narrowband bearerCapability, either
-- DSS1 (ES 300 403-1) or the ISUP User Service Information (ITU-T Recommendation Q.763)
-- encoding can be used. Refer
-- to the ITU-T Recommendation Q.763 Transmission Medium Requirement parameter for tmr encoding.


BothwayThroughConnectionInd::= ENUMERATED {
    bothwayPathRequired    (0),
    bothwayPathNotRequired (1)
    }
```
-- The default is bothwayPathRequired.

```
CAI-GSM0224                      ::= SEQUENCE {
    e1                           [0] INTEGER (0..8191)            OPTIONAL,
    e2                           [1] INTEGER (0..8191)            OPTIONAL,
    e3                           [2] INTEGER (0..8191)            OPTIONAL,
    e4                           [3] INTEGER (0..8191)            OPTIONAL,
    e5                           [4] INTEGER (0..8191)            OPTIONAL,
    e6                           [5] INTEGER (0..8191)            OPTIONAL,
    e7                           [6] INTEGER (0..8191)            OPTIONAL,

...
    }
-- Support for CAMEL
-- Indicates Charge Advice Information to the Mobile Station. For information regarding
-- parameter usage, refer to TS 122 024.

CalledDirectoryNumber  {B2: b2}::= OCTET STRING (SIZE
                    (b2.&minCalledDirectoryNumberLength..
                 b2.&maxCalledDirectoryNumberLength))
-- Indicates the Called Directory Number.
-- Refer to ITU-T Recommendation Q.763  'Called Directory Number' for encoding.


CalledPartyBCDNumber {B2: b2} ::= OCTET STRING (SIZE
        (b2.&minCalledPartyBCDNumberLength.. b2.&maxCalledPartyBCDNumberLength))
-- Indicates the Called Party Number, including service selection information.
-- Refer to TS 124 008  for encoding.
-- This data type carries only the "type of number", "numbering plan identification" and
-- "number digit" fields defined in TS 124 008;
-- it does not carry the "called party BCD number IEI" or
-- "length of called party BCD number contents".

CalledPartyNumber  {B2: b2}::= OCTET STRING (SIZE
                    (b2.&minCalledPartyNumberLength..
                 b2.&maxCalledPartyNumberLength))
-- Indicates the Called Party Number. Refer to ITU-T Recommendation Q.763 for encoding.


CallingGeodeticLocation {B2: b2}::= OCTET STRING ( SIZE
(b2.&minCallingGeodeticLocationLength..b2.&maxCallingGeodeticLocationLength))
-- The coding of this parameter is based on the appropriate mapping with the
-- ISUP parameter Calling Geodetic Location.
-- Refer to ITU-T Recommendation Q.763 for encoding.

CallingPartyBusinessGroupID::= OCTET STRING
-- Indicates the business group of the calling party. The value of this octet string is network
-- operator specific.
-- Its content is network operator specific.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this parameter
-- (after the first tag and length
-- information) is the BER encoding of the defined ASN.1 internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.

CallingPartyNumber  {B2: b2}::= OCTET STRING (SIZE (
                    b2.&minCallingPartyNumberLength..
                      b2.&maxCallingPartyNumberLength))
-- Indicates the Calling Party Number.  Refer to ITU-T Recommendation Q.763 for encoding.
```

```
CallingPartysCategory::= OCTET STRING (SIZE(1))
-- Indicates the type of calling party (e.g. operator, payphone, ordinary subscriber).
-- Refer to ITU-T Recommendation Q.763 for encoding


CallReference {B2: b2}::= OCTET STRING (SIZE(1..b2.&maxCallReferenceLength))
-- The coding of this parameter is network specific.
-- A possible coding is the ITU-T Recommendation Q.763 call reference
-- but other encoding schemes are possible.


CAMEL-AChBillingChargingCharacteristics {B1: b1}::= CHOICE {
    timeDurationCharging              [0] SEQUENCE {
        maxCallPeriodDuration             [0] INTEGER (1..864000),
        releaseIfdurationExceeded         [1] BOOLEAN DEFAULT FALSE,
        tariffSwitchInterval              [2] INTEGER (1..86400)          OPTIONAL,
        tone                              [3] BOOLEAN DEFAULT FALSE,
        extensions                        [4] Extensions  {b1} OPTIONAL,
        ...
    }
}
-- tariffSwitchInterval is measured in 1 second units.
-- maxCallPeriodDuration is measured in100 millisecond units


CAMEL-CallResult {B1: b1}       ::= CHOICE {
    timeDurationChargingResult        [0] SEQUENCE {
        partyToCharge                     [0] ReceivingSideID,
        timeInformation                   [1] TimeInformation,
        callActive                        [2] BOOLEAN              DEFAULT TRUE,
        callReleasedAtTcpExpiry           [3] NULL                 OPTIONAL,
        extensions                        [4] Extensions {b1}      OPTIONAL,
        ...
    }
}


CAMEL-FCIBillingChargingCharacteristics {B2: b2}::= CHOICE{
    fCIBCCCAMELsequence1           [0] SEQUENCE {
        freeFormatData                 [0] OCTET STRING (SIZE
(b2.&minCamelFCIBillingChargingDataLength..
    b2.&maxCamelFCIBillingChargingDataLength)),
        partyToCharge                  [1] SendingSideID
                                                DEFAULT sendingSideID: leg1,
        ...
    }
}


CAMEL-SCIBillingChargingCharacteristics::= CHOICE {
    aOCBeforeAnswer                        [0] AOCBeforeAnswer,
    aOCAfterAnswer                         [1] AOCSubsequent
    }


GlobalCallReference {B2: b2}::= OCTET STRING (SIZE(1..b2.&maxGlobalCallReferenceLength))
-- The coding of this parameter is defined in annex D.


CallResult {B1: b1, B2: b2}          ::= CHOICE {
    callResult  OCTET STRING (SIZE (b2.&minCallResultcs1Length..
                              b2.&maxCallResultcs1Length)),
-- This parameter provides the SCF with the charging related information previously requested
-- using the ApplyCharging operation.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this paramter
-- (after the first tag and length
-- information) is the BER encoding of the defined ASN.1 internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.
    callSupervisionResult  [51]        SEQUENCE {
        callResult                        [0] OCTET STRING (SIZE (b2.&minCallResultcs1Length..
                                          b2.&maxCallResultcs1Length))   OPTIONAL,
        reportConditionInformation  [1] ReportConditionInformation,
        aChChargingAddress          [2] AChChargingAddress {b2} DEFAULT legID:receivingSideID:leg1,
        supervisionResult           [3] CHOICE {
                    cost              [1] UsedCost,
                    time              [2] UsedTime,
                    pulses            [3] UsedPulses
        },
```

```
        minLimitNeeded              [4] CHOICE {
                        cost        [1] NeededCost,
                        pulses      [2] NeededPulses
                },
        extensions                  [5] Extensions {b1}  OPTIONAL,
        ...
        }
    }
```

```
-- This parameter provides the SCF with the charging related information previously requested
-- using the ApplyCharging operation.
-- Examples of charging related information to be provided by the SSF may be: bulk counter values,
-- costs, tariff change and time of change, time stamps, durations, etc.
-- Examples of conditions on which the charging related information are to be reported may be:
-- threshold value reached, timer expiration, tariff change, end of connection configuration, etc.
-- CAMEL:
-- CallResult {PARAMETERS-BOUND: bound}::= OCTET STRING (SIZE (b2.&minCallResultLength..
--  b2.&maxCallResultLength))
--  (CONSTRAINED BY {
-- shall be the result of the BER-encoded value of type --
--  CAMEL-CallResult {bound}})
-- The violation of the UserDefinedConstraint shall be handled as an ASN.1 syntax error.
-- This parameter provides the gsmSCF with the charging related information previously requested
-- using the ApplyCharging operation. This shall include the partyToCharge parameter as
-- received in the related ApplyCharging operation to correlate the result to the request
-- The value of the CallResult of type OCTET STRING carries
-- a value of the ASN.1 data type: CAMEL-CallResult.
-- The normal encoding rules are used to encode this value.
```

```
CallSegmentID {B2: b2}::= INTEGER (1..b2.&numOfCSs)
```

```
CallSupervision {B2: b2}    ::= SEQUENCE {
    supervisionMethod           [1] SupervisionMethod,
    warningBeforeLimitReached   [2] WarningBeforeLimitReached           OPTIONAL,
    releaseWhenLimitReached     [3] ReleaseWhenLimitReached {b2}        OPTIONAL,
    reportConditions            [4] ReportConditions                    OPTIONAL,
    ...
    }
```

```
Carrier {B2: b2}::= OCTET STRING (SIZE (b2.&minCarrierLength..
                    b2.&maxCarrierLength))
```

```
-- Contains the carrier selection field (first octet) followed by either  Carrier ID information
-- (option 1 (e.g. North America (na)) ), or the Transit
-- Network selection information (option 2 (e.g. Europa)), depending on the network
-- in which the switch is located..

-- In both cases, the Carrier selection is one octet and is encoded as:
-- 00000000    No indication
-- 00000001    Selected carrier identification code (CIC) pre subscribed and not
--             input by calling party
-- 00000010    Selected carrier identification code (CIC) pre subscribed and input by calling party
-- 00000011    Selected carrier identification code (CIC) pre subscribed, no indication of
--             whether input by calling party (undetermined)
-- 00000100    Selected carrier identification code (CIC) not pre subscribed and input by
--             calling party
-- 00000101
--  to         Spare
-- 11111110
-- 11111111    Reserved

-- Refer to ITU-T Recommendation Q.763 for encoding of Transit Network Selection.

-- Refer to ANSI ISUP T.113 for encoding of na carrier ID information
```

```
Cause  {B2: b2}::= OCTET STRING (SIZE (minCauseLength..
                        b2.&maxCauseLength))
-- Indicates the cause for interface related information.
-- Refer to the ITU-T Recommendation Q.763 Cause  parameter for encoding
-- For the use of cause and location values refer to ITU-T Recommendation Q.850.
```

```
CCSS::=BOOLEAN
-- Used by the SSF to indicate CCSS (Call Completion on Service Set-up) if set to "True"
-- to the SCF, i.e.
-- that the current call is due a special procedure  (CCBS or CCNR).
-- The value TRUE corresponds with the ITU-T Recommendation Q.763 CCSS parameter
-- "CCSS call indicator" value 1 (CCSS call).
-- The value FALSE corresponds with the ITU-T Recommendation Q.763 CCSS parameter
-- "CCSS call indicator" value 0 (no indication).


CGEncountered::= ENUMERATED {
    noCGencountered     (0),
    manualCGencountered (1),
    sCPOverload         (2)
    }
-- Indicates the type of automatic call gapping encountered, if any.


ChargedParty                 ::= ENUMERATED {
    callingPartyNumber  (0),
    calledINNumber      (1),
    connectedNumber     (2),
    translatedNumber    (3),
    specialINNumber     (4),
    ...
    }
-- The TranslatedNumber is on the ISUP named CalledPartyNumber
-- (e.g. translated number after Connect operation).


ChargedPartyNumber {B2: b2} ::= OCTET STRING (SIZE(b2.&minChargedPartyNumberLength..
                              b2.&maxChargedPartyNumberLength) )
-- Indicates the number that identifies the entity or party to be charged for the call.
-- The coding of this parameter is network specific.
-- A possible coding is the ITU-T Recommendation Q.763 called party number
-- but other encoding schemes are possible.


ChargeNumber  {B2: b2}::= LocationNumber {b2}
-- This parameter is only used for CAMEL. It uniquely identifies the chargeable number
-- of a call sent into a North American
-- long distance carrier. It transports the ChargeNumber Parameter Field
-- as defined in ANSI ISUP T1.113. This provides 1 octet for the nature of address indicator field,
-- plus 1 octet for a numbering plan field, plus up to 5 octets for the address signal
-- (up to 10  digits). It is used in the context of CAMEL.
--  The Charge Number in ANSI T1.113 normally contains a 10 digit national number within the North
--  American Numbering Plan (NANP); longer (e.g. international) charge numbers are not supported in
--  T1.113-
-- Information indicating the chargeable number for the call and consisting of  the
-- odd/even indicator, nature of address indicator, numbering plan indicator,
-- and address signals.
-- Uses the LocationNumber format for encoding which is based on the ITU-T Recommendation Q.763
Location Number format

--ChargeUnitTimeIntervalType


ChargingAddress    {B2: b2}       ::= CHOICE {
    leg                  [2] LegID,
    srfCallSegment       [50] CallSegmentID {b2}
    }

ChargingControlType                ::= SEQUENCE {
    tariffInfoFromSuccExchangeSCI     [0]     TariffInfoFromSuccExchangeSCI,
    sSFdetermination                  [1]     SSFdetermination DEFAULT noDetermination,
    ...
    }

ChargingEvent {B2: b2}  ::= SEQUENCE {
    eventTypeCharging     [0] EventTypeCharging {b2},
    monitorMode           [1] MonitorMode,
    legID                 [2] LegID                OPTIONAL,
    eventTypeTariff       [50] EventTypeTariff   OPTIONAL,
    ...
    }
-- This parameter indicates the charging event type and corresponding monitor mode and LedID
```

```
CNInfo {B2: b2}::= OCTET STRING (SIZE(1.. b2.&maxCNInfoLength))
-- Refer to ITU-T Recommendation Q.765.1 NNI specific information to be transported in the
-- Application Transport Parameter (APP).


-- The NNI specific information for the VPN application is carried within the APP.
-- Only the CNID (Corporate Telecommunications Network Identifier) indicator is significant
-- for INAP (bits 5 and 6).
-- The CNID indicator is followed by optionally (if included) the CNID length and CNID value.
-- CommunicationChargeCurrency Type


ConnectedNumberTreatmentInd::= ENUMERATED {
          noINImpact                      (0),
          presentationRestricted          (1),
          presentCalledINNumber           (2),
          presentCalledINNumberRestricted (3)
          }
-- The default is presentCalledINNumber.

ControlType::= ENUMERATED {
    sCPOverloaded       (0),
    manuallyInitiated   (1),
    destinationOverload (2)
    }

CompoundCriteria {B2: b2}   ::= SEQUENCE {
        basicGapCriteria        [0] BasicGapCriteria {b2},
        scfID                   [1] ScfID {b2} OPTIONAL,
        ...
                              }
-- If scfID parameter is not available the call gapping is not dedicated to a specific SCF.


CorrelationID {B2: b2}::= Digits {b2}
-- used by SCF for correlation with a previous operation.
-- refer to ITU-T Recommendation Q.763 Generic Digits for encoding from SCF to SSF.
-- refer to ITU-T Recommendation Q.763 Generic Number for encoding from SRF/SSF to SCF.


CounterAndValue::= SEQUENCE {
        counterID      [0] CounterID,
        counterValue   [1] Integer4,
        ...
        }

CounterID::= INTEGER (0..99)
-- Indicates the counters to be incremented.
-- The counterIDs are addressed by using the last digits of the dialed number.

CountersValue::= SEQUENCE SIZE(0..numOfCounters) OF CounterAndValue

CreateOrRemoveIndicator::= ENUMERATED {
          create (0),
          remove (1)
          }

-- Indicates the creation or removal of a TDP-R.

CSAID {B2: b2}::= INTEGER (1..b2.&numOfCSAs)
-- Indicates the SSF CSA identifier


-- Currency Type
```

```
Currency::= ENUMERATED {
    noIndication         (0),
    australianDollar     (1),
    austrianSchilling    (2),
    belgianFranc         (3),
    britishPound         (4),
    czechKoruna          (5),
    danishKrone          (6),
    dutchGuilder         (7),
    euro                 (8),
    finnishMarkka        (9),
    frenchFranc          (10),
    germanMark           (11),
    greekDrachma         (12),
    hungarianForint      (13),
    irishPunt            (14),
    italianLira          (15),
    japaneseYen          (16),
    luxembourgian-Franc  (17),
    norwegianKrone       (18),
    polishZloty          (19),
    portugeseEscudo      (20),
    russianRouble        (21),
    slovakKoruna         (22),
    spanishPeseta        (23),
    swedishKrone         (24),
    swissFranc           (25),
    turkishLira          (26),
    uSDollar             (27),
    ...}
```

-- CurrencyFactor Type

CurrencyFactor::= INTEGER (0..999999)

-- Value 0 indicates "no charge".

CurrencyScale::= INTEGER (-7..3)

-- The actual value for currency scale is given by 10^X, where x is the value of the CurrencyScale.
-- The coding of CurrencyScale is as follows, all other values are spare:
-- -7 (249): 0,0000001
-- -6 (250): 0,000001
-- -5 (251): 0,00001
-- -4 (252): 0,0001
-- -3 (253): 0,001
-- -2 (254): 0,01
-- -1 (255): 0,1
-- 0   : 1
-- 1   : 10
-- 2   : 100
-- 3   : 1000

CutAndPaste::= INTEGER (0..22)
-- Indicates the number of leading digits to be deleted (cut) and to paste remaining dialed digits.

DateAndTime::= OCTET STRING (SIZE(6..7))
-- Indicates, amongst others, the start time and stop time for activate service filtering.
-- Coded as YYMMDDHHMMSS (option 1) or YYYYMMDDHHSS (option 2) with each digit coded BCD

-- OPTION 1 (Size 6):
-- The first octet contains YY and the remaining items are sequenced following
-- For example, 1998 September [30]th, 12:15:01 would be encoded as:
-- Bits   HGFE   DCBA
-- leading octet   8   9
--   9   0
--   0   3
--   2   1
--   5   1
--   1   0
-- The 2 digit value
--   representing a Year shall be interpreted as follows
-- If the two-digits value is 00 through 49 inclusive, it shall be interpreted as representing
-- year 2000 through 2049.
-- If the two-digits value is 50 through 99 inclusive, it shall be interpreted as representing
-- year 1950 through 1999.

```
-- Option 2 (Size 7):
-- Support for CAMEL.
-- The year digit indicating millenium occupies bits
-- 0-3 of the first octet, and the year digit indicating century occupies bits
-- 4-7 of the first octet.
-- The year digit indicating decade occupies bits 0-3 of the second octet,
-- whilst the digit indicating the year within the decade occupies bits 4-7 of
-- the second octet.
-- The most significant month digit occupies bits 0-3 of the third octet,
-- and the least  significant month digit occupies bits 4-7 of the third octet.
-- The most significant day digit occupies bits 0-3 of the fourth octet,
-- and the least significant day digit occupies bits 4-7 of the fourth octet.
-- The most significant hours digit occupies bits 0-3 of the fifth octet,
-- and the least significant digit occupies bits 4-7 of the fifth octet.
-- The most significant minutes digit occupies bits 0-3 of the sixth octet,
-- and the least significant minutes digit occupies bits 4-7 of the sixth octet.
-- The most significant seconds digit occupies bits 0-3 of the seventh octet,
-- and the least seconds significant digit occupies bits 4-7 of the seventh octet.
-- For the encoding of digits in an octet, refer to the timeAndtimezone parameter.


DefaultFaultHandling {B1: b1, B2: b2, B3: b3}::= SEQUENCE{
    action        [0]           ENUMERATED  {
        resumeCallProcessing   (0),
        releaseCall            (1),
        ...
        },
    treatment   [1] GapTreatment {b1, b2,b3}  OPTIONAL,
    ...
    }
-- The data type GapTreatment was reused from the existing types to avoid the definition
-- of a new one.

DestinationRoutingAddress {B2: b2}::=
    SEQUENCE SIZE(1) OF CalledPartyNumber {b2}
-- Indicates the destination address for routeing.

Digits {B2: b2}::=
    OCTET STRING (SIZE (b2.&minDigitsLength..b2.&maxDigitsLength))

-- Indicates the address signalling digits.
--  Refer to the ITU-T Recommendation Q.763 Generic Number and Generic Digits parameter for
encoding.
-- The digits may also include the '#', '*', a, b and c digits.
-- The coding of the subfield's 'NumberQualifier' in Generic Number and 'TypeOfDigits' in
-- Generic Digits are irrelevant to the INAP, the ASN.1 tags are sufficient to
-- identify the parameter.
-- The ISUP format does not allow to exclude these subfields, therefore the value
-- is network operator specific.
-- The following parameters should use Generic Number
-- Additional Calling Party Number, CorrelationID for AssistRequestInstructions,
-- AssistingSSPIPRoutingAddress for EstablishTemporaryConnection,
-- calledAddressValue for all occurrences, callingAddressValue for all occurrences
-- The following parameters should use Generic Digits: all
-- other CorrelationID occurrences,, lineID for ResourceID type, digitResponse for
-- ReceivedInformationArg, iNServiceControlLow / iNServiceControlHigh for
-- MidCallInfoType, iNServiceControlCode for MidCallInfo.

DisplayInformation {B2: b2}::= IA5String (SIZE (b2.&minDisplayInformationLength..
    b2.&maxDisplayInformationLength))
-- Indicates the display information

DpSpecificCriteria {B2: b2}::= CHOICE {
        numberOfDigits         [0] NumberOfDigits,
        applicationTimer       [1] ApplicationTimer,
        midCallControlInfo     [2] MidCallControlInfo {b2},
        numberOfDigitsTwo      [3] SEQUENCE {
            requestedNumberOfDigits [0] NumberOfDigits,
            minNumberOfDigits       [1] NumberOfDigits OPTIONAL,
            ...
            }
        }
```

```
-- The SCF may specify the number of digits to be collected by the SSF for the CollectedInfo event
-- When all digits are collected, the SSF reports the event to the SCF
-- The SCF may set a timer in the SSF for the No Answer event. If the user does not answer the call
-- within the allotted time, the SSF reports the event to the SCF
-- The SCF may specify the number of digits to be collected by the SSF for the
-- CollecteInfo event and hereby specify a minimum number of digits to be collected in case
-- the exact number of digits is unknown to the SCF, but a  report is desired in case
-- of complete number is determined before the requested number of digits has been collected.

Duration::= INTEGER (-2..86400)
-- Values are seconds
-- special meaning applies for -2, -1 and 0 values as described for the concerned
-- operation procedures.

Entry::= CHOICE {
    agreements       [0] OBJECT IDENTIFIER,
    networkSpecific [1] Integer4
    }

EventSpecificInformationBCSM {B2: b2}::= CHOICE {
    collectedInfoSpecificInfo      [0] SEQUENCE {
            calledPartynumber      [0] CalledPartyNumber {b2},
            ...
            },
    analysedInfoSpecificInfo       [1] SEQUENCE {
            calledPartynumber      [0] CalledPartyNumber {b2},
            ...
            },
    routeSelectFailureSpecificInfo [2] SEQUENCE {
            failureCause           [0] Cause {b2} OPTIONAL,
            ...
            },
    oCalledPartyBusySpecificInfo   [3] SEQUENCE {
            busyCause              [0] Cause {b2} OPTIONAL,
            ...
            },
    oNoAnswerSpecificInfo          [4] SEQUENCE {
            oNoAnswerCause         [0] Cause {b2} OPTIONAL,
                       ...
            },
    oAnswerSpecificInfo            [5] SEQUENCE {

            backwardGVNS           [0] BackwardGVNS {b2} OPTIONAL,
            destinationAddress     [50]   CalledPartyNumber {b2} OPTIONAL,
-- CAMEL support
            or-Call                [51]   NULL                  OPTIONAL,
-- CAMEL support
            forwardedCall          [52]   NULL                  OPTIONAL,
-- CAMEL support
    ...
            },
    oMidCallSpecificInfo           [6] SEQUENCE {
            connectTime            [0] Integer4 OPTIONAL,
            oMidCallInfo           [1] MidCallInfo {b2} OPTIONAL,
            ...
            },
    oDisconnectSpecificInfo    [7] SEQUENCE {
            releaseCause           [0] Cause {b2} OPTIONAL,
            connectTime            [1] Integer4 OPTIONAL,
            ...
            },
    tBusySpecificInfo              [8] SEQUENCE {
            busyCause              [0] Cause {b2} OPTIONAL,
            callForwarded          [50] NULL              OPTIONAL,
-- CAMEL suport
            ...
            },
    tNoAnswerSpecificInfo          [9] SEQUENCE {
            tNoAnswerCause         [0] Cause {b2} OPTIONAL,
            callForwarded          [50] NULL      OPTIONAL,
-- CAMEL support
            ...
            },
```

```
        tAnswerSpecificInfo         [10] SEQUENCE {

                destinationAddress          [50]    CalledPartyNumber {b2}  OPTIONAL,
-- CAMEL support
                or-Call                     [51]    NULL                    OPTIONAL,
-- CAMEL support
                forwardedCall               [52]    NULL                    OPTIONAL,
-- CAMEL support
                ...
                },
        tMidCallSpecificInfo        [11] SEQUENCE {
                connectTime                 [0] Integer4 OPTIONAL,
                tMidCallInfo                [1] MidCallInfo {b2} OPTIONAL,
                ...
                },
        tDisconnectSpecificInfo     [12] SEQUENCE {
                releaseCause                [0] Cause {b2} OPTIONAL,
                connectTime                 [1] Integer4 OPTIONAL,
                ...
                },
        oTermSeizedSpecificInfo     [13] SEQUENCE {
                -- no specific info defined


                ...
                },
        oSuspendSpecificInfo        [14] SEQUENCE {
                -- no specific info defined
                ...
                },
        tSuspendSpecificInfo        [15] SEQUENCE {
                -- no specific info defined
                ...
                },
        origAttemptAuthorizedSpecificInfo   [16] SEQUENCE {
                -- no specific info defined
                ...
                },
        oReAnswerSpecificInfo       [17] SEQUENCE {
                -- no specific info defined
                ...
                },
        tReAnswerSpecificInfo       [18] SEQUENCE {
                -- no specific info defined
                ...
                },
        facilitySelectedAndAvailableSpecificInfo    [19] SEQUENCE {
                -- no specific info defined
                ...
                },
        callAcceptedSpecificInfo    [20] SEQUENCE {
                -- no specific info defined
                ...
                },
        oAbandonSpecificInfo        [21] SEQUENCE {
                abandonCause                [0] Cause {b2} OPTIONAL,
                ...
                },
        tAbandonSpecificInfo        [22] SEQUENCE {
                abandonCause                [0] Cause {b2} OPTIONAL,
                ...
                },
        authorizeRouteFailureSpecificInfo   [23] SEQUENCE {
                authorizeRouteFailureCause      [0] Cause {b2} OPTIONAL,
                ...
                },
        terminationAttemptAuthorizedSpecificInfo    [24] SEQUENCE {
                -- no specific info defined
                ...
                },
        orininationAttemptDeniedSpecificInfo        [25] SEQUENCE {
                originationDeniedCause              [0] Cause {b2} OPTIONAL,
                ...
                },
```

```
        terminationAttemptDeniedSpecificInfo        [26] SEQUENCE {
                terminationDeniedCause               [0] Cause {b2} OPTIONAL,
                ...
                }

        }
-- Indicates the call related information specific to the event.
-- The unit for the connectTime is 100 milliseconds


EventSpecificInformationCharging {B2: b2}::= OCTET STRING (SIZE
                        (b2.&minEventSpecificInformationChargingLength..
                        b2.&maxEventSpecificInformationChargingLength))
-- defined by network operator.
-- Its content is network signalling/operator specific.
-- Indicates the charging related information specific to the event.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this parameter (after the first tag and length
-- information) is the BER encoding of the defined ASN.1 internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.


EventTypeBCSM::= ENUMERATED {
    origAttemptAuthorized        (1),
    collectedInfo                (2),
    analysedInformation          (3),
    routeSelectFailure           (4),
    oCalledPartyBusy             (5),
    oNoAnswer                    (6),
    oAnswer                      (7),
    oMidCall                     (8),
    oDisconnect                  (9),
    oAbandon                     (10),
    termAttemptAuthorized        (12),
    tBusy                        (13),
    tNoAnswer                    (14),
    tAnswer                      (15),
    tMidCall                     (16),
    tDisconnect                  (17),
    tAbandon                     (18),
    oTermSeized                  (19),
    oSuspend                     (20),
    tSuspend                     (21),
    origAttempt                  (22),
    termAttempt                  (23),
    oReAnswer                    (24),
    tReAnswer                    (25),
    facilitySelectedAndAvailable(26),
    callAccepted                 (27),
    authorizeRouteFailure        (28),
    originationAttemptDenied     (29),
    terminationAttemptDenied     (30)
    }

-- Indicates the name of the BCSM detection point event.
-- The value range 100 - 127 is reserved.

EventTypeCharging {B2: b2}::= OCTET STRING (SIZE
                        (b2.&minEventTypeChargingLength..
                         b2.&maxEventTypeChargingLength))
-- This parameter indicates the charging event type..
-- Its content is network signalling / operator specific.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this parameter (after the first tag and length
-- information) is the BER encoding of the defined ASN.1 internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.


EventTypeTariff              ::= ENUMERATED {
    chargingTariffInformation        (0),
    addOnchargingInformation         (1),
    chargingAcknowledgementInformation (2),
    chargingAcknowledgeTimerExpired   (3),
    startCharging                    (4),
    stopCharging                     (5)
    }
```

```
FacilityGroup::= CHOICE {
    trunkGroupID        [0] INTEGER,
    privateFacilityID   [1] INTEGER,
    huntGroup           [2] OCTET STRING,
    routeIndex          [3] OCTET STRING
    }
-- Indicates the particular group of facilities to route the call.
-- huntGroup and routeIndex are encoded as
-- network operator specific.
-- The internal structure of huntGroup and routeIndex parameters can be defined using ASN.1
-- and the related Basic Encoding Rules (BER). In such a case the value of this parameter
-- (after the first tag and length information) is the BER encoding of the defined ASN.1
-- internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.


FCIBillingChargingCharacteristics {B1: b1, B2: b2}::= CHOICE {
    fCIBCCcs1   OCTET STRING (SIZE (b2.&minFCIBCCcs1Length..
                                    b2.&maxFCIBCCcs1Length)),
-- Its content is network operator specific.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this parameter (after the first tag and
--length information) is the BER encoding of the defined ASN.1 internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.


    fCIBCCsequence       [51] SEQUENCE {


        fCIBCC                   [0] OCTET STRING (SIZE
(b2.&minFCIBCCcs2Length..b2.&maxFCIBCCcs2Length))  OPTIONAL,
-- Its content is network operator specific.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this parameter (after the first tag
-- and length information) is the BER encoding of the defined ASN.1 internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.
        chargeMessageFromSCF      [1] CHOICE {
            sCFChargingTariff              [0] ChargingTariffInformation,
            sCFAddOnCharge                 [1] AddOnChargingInformation,
            sSFDeterminedTariff            [2]  NULL
            }
        OPTIONAL,


        tariffFromSuccExchange    [2] TariffFromSuccExchange DEFAULT notTakenIntoAccount,
        freeFormatData            [3] OCTET STRING (SIZE
(b2.&minFormatDataLength..b2.&maxFormatDataLength)) OPTIONAL,


        chargingAddress           [4] ChargingAddress {b2}    DEFAULT leg:sendingSideID:leg2,
        partyToChargeIdentifier   [5] PartyToCharge {b2}  OPTIONAL,
        iNRecordIndicators        [6] INRecordIndicators OPTIONAL,
-- The parameter should always be included.
        extensions                [7] Extensions {b1}  OPTIONAL,
        ...
        }


    }
-- This FCIBillingChargingCharacteristics parameter indicates the billing and/or
-- charging characteristics.
-- Its content is partly network operator specific.
-- CAMEL:
-- FCIBillingChargingCharacteristics {PARAMETERS-BOUND: bound}::= OCTET STRING (SIZE
--  (bound.&minFCIBillingChargingLength..bound.&maxFCIBillingChargingLength))
--  (CONSTRAINED BY {
-- shall be the result of the BER-encoded value of type --
--  CAMEL-FCIBillingChargingCharacteristics {bound}})
-- This parameter indicates the billing and/or charging characteristics.
-- The violation of the UserDefinedConstraint shall be handled as an ASN.1 syntax error.
-- The value of the FCIBillingChargingCharacteristics of type OCTET STRING carries
-- a value of the ASN.1 data type: CAMEL-FCIBillingChargingCharacteristics.
-- The normal encoding rules are used to encode this value.
```

```
FilteredCallTreatment {B1: b1, B2: b2, B3: b3}::= SEQUENCE {
    sFBillingChargingCharacteristics    [0] SFBillingChargingCharacteristics {b2}   OPTIONAL,
    informationToSend                   [1] InformationToSend {b1, b2, b3}  OPTIONAL,
    maximumNumberOfCounters             [2] MaximumNumberOfCounters OPTIONAL,
    releaseCause                        [3] Cause {b2}  OPTIONAL,
    sFTariffMessage                     [50] CHOICE {
        crgt                               [0] ChargingTariffInformation}  OPTIONAL,

    ...
    }
-- If releaseCause is not present, the default value is the same as the ISUP cause value decimal 31.
-- If informationToSend is present, the call will be released after the end of the announcement
-- with the indicated or default releaseCause.
-- If maximumNumberOfCounters is not present, ServiceFilteringResponse will be sent with
-- CountersValue::= SEQUENCE SIZE (0) OF CountersAndValue.


FilteringCharacteristics::= CHOICE {
    interval        [0] INTEGER (-1..32000),
    numberOfCalls   [1] Integer4
    }
-- The interval is specified in seconds.


FilteringCriteria {B2: b2}::= CHOICE {
    serviceKey          [2] ServiceKey,
    addressAndService   [30] SEQUENCE {
            calledAddressValue    [0] Digits {b2},
            serviceKey            [1] ServiceKey,
            callingAddressValue   [2] Digits {b2} OPTIONAL,
            locationNumber        [3] LocationNumber {b2} OPTIONAL,
            ...
            }
    }
-- In case calledAddressValue is specified, the numbers to be filtered are from calledAddressValue
-- up to and including calledAddressValue + maximumNumberOfCounters-1.
-- The last two digits of calledAddressvalue cannot exceed 100-maximumNumberOfCounters.
-- For encoding of the digits, refer to ITU-T Recommendation Q.763 for Generic Number.


FilteringTimeOut::= CHOICE {
    duration        [0] Duration,
    stopTime        [1] DateAndTime
    }
-- Indicates the maximum duration of the filtering.
-- When the timer expires, a ServiceFilteringResponse is sent to the SCF.


ForwardCallIndicators::= OCTET STRING (SIZE(2))
-- Indicates the Forward Call Indicators. Refer to ITU-T Recommendation Q.763 for encoding


ForwardGVNS {B2: b2}::= OCTET STRING (SIZE(
            b2.&minForwardGVNSLength..
            b2.&maxForwardGVNSLength))
-- Indicates the GVNS Forward information. Refer to ITU-T Recommendation Q.735 for encoding.


ForwardServiceInteractionInd::= SEQUENCE {
    conferenceTreatmentIndicator        [1] OCTET STRING (SIZE(1))  OPTIONAL,
    -- acceptConferenceRequest              'xxxx xx01'B
    -- rejectConferenceRequest              'xxxx xx10'B
    -- network default is accept conference request.

    callDiversionTreatmentIndicator     [2] OCTET STRING (SIZE(1))  OPTIONAL,
    -- callDiversionAllowed                 'xxxx xx01'B
    -- callDiversionNotAllowed              'xxxx xx10'B
    -- network default is Call Diversion allowed.

    callOfferingTreatmentIndicator      [3] OCTET STRING (SIZE(1))  OPTIONAL,
    -- callOfferingNotAllowed               'xxxx xx01'B,
    -- callOfferingAllowed                  'xxxx xx10'B
    -- callOfferingNoINImpact               'xxxx x100'B
    -- indicates if call offering is "allowed", "not allowed" or "no impact by IN".
    -- network default is Call Offering not allowed.

    callingPartyRestrictionIndicator    [4] OCTET STRING (SIZE(1))  OPTIONAL,
    -- noINImpact                           'xxxx xx01'B,
    -- presentationRestricted               'xxxx xx10'B
    -- network default is noINImpact
```

```
    callWaitingTreatmentIndicator      [5] OCTET STRING (SIZE(1))  OPTIONAL,
    -- callWaitingAllowed                            'xxxx xx01'B,
    -- callWaitingNotAllowed                         'xxxx xx10'B
    -- network default is Call Waiting allowed

    holdTreatmentIndicator             [6] OCTET STRING (SIZE(1))  OPTIONAL,
    -- acceptHoldRequest   'xxxx xx01'B
    -- rejectHoldRequest   'xxxx xx10'B
    -- network default is accept hold request

    ectTreatmentIndicator              [7] OCTET STRING (SIZE(1))  OPTIONAL,
    -- acceptEctRequest 'xxxx xx01'B
    -- rejectEctRequest 'xxxx xx10'B
    -- network default is accept ect request
    ...
}

--The forwardServiceInteractionInd parameter is applicable to IDP, CON, CWA and ICA operations.

GapCriteria {B2: b2}::= CHOICE {
    basicGapCriteria        BasicGapCriteria {b2},
    compoundGapCriteria     CompoundCriteria {b2}
    }


GapOnService::= SEQUENCE {
    serviceKey [0] ServiceKey,
    ...
    }
GapIndicators::= SEQUENCE {
    duration    [0] Duration,
    gapInterval [1] Interval,
    ...
    }
-- Indicates the gapping characteristics.

GapTreatment  {B1: b1, B2: b2, B3: b3}::= CHOICE {
    informationToSend   [0] InformationToSend {b1, b2, b3},
    releaseCause        [1] Cause {b2},
    both                [2] SEQUENCE {
            informationToSend  [0] InformationToSend {b1, b2, b3},
            releaseCause       [1] Cause {b2},
        ...
            }
    }
-- The default value for Cause is the same as in ISUP.

GenericName  {B2: b2}::= OCTET STRING (SIZE(
            b2.&minGenericNameLength..
            b2.&maxGenericNameLength))
-- Refer to ITU-T Recommendation Q.931 Display Information parameter for encoding.


GenericNumber  {B2: b2}::= OCTET STRING (SIZE(
            b2.&minGenericNumberLength..
            b2.&maxGenericNumberLength))
-- Refer to ITU-T Recommendation Q.763 Generic Number for encoding.

GenericNumbers {B2: b2}::=
    SET SIZE(1..b2.&numOfGenericNumbers) OF GenericNumber {b2}

HighLayerCompatibility::= OCTET STRING (SIZE (highLayerCompatibilityLength))
-- Indicates the teleservice. For encoding, DSS1 (ITU-T Recommendation Q.931) is used.

INRecordIndicators              ::= SEQUENCE {
    iNRecordAction              [0] INRecordAction DEFAULT generateCallRecord,
    hotBillingRequired          [1] BOOLEAN DEFAULT FALSE,
    ...
    }

INRecordAction                  ::= ENUMERATED {
    generateCallRecord          (1),
    appendDataToCallRecord      (2),
    overwriteDataForCallRecord  (3),
    noDataForCallRecord         (4),
    ...
    }
```

```
initialCallSegment INTEGER::= 1

InitialDPArgExtension              ::= SEQUENCE {
    gmscAddress                        [1] ISDN-AddressString     OPTIONAL,
    ...
}

INprofile {B1: b1, B2:b2} ::=SEQUENCE {
        actionOnProfile     [0] ActionOnProfile,
        tDPIdentifier       [1] TDPIdentifier {b2},
        dPName              [2] EventTypeBCSM  OPTIONAL,
        extensions          [3] Extensions {b1}  OPTIONAL,
        ...
        }

INServiceCompatibilityIndication {B2: b2}::=
    SEQUENCE SIZE (1..b2.&numOfInServiceCompatibilityIndLength) OF Entry

INServiceCompatibilityResponse  ::= Entry

Interval::= INTEGER (-1..60000)
-- Units are in milliseconds. A -1 value denotes infinite.

IPAvailable {B2: b2}::= OCTET STRING (SIZE (
        b2.&minIPAvailableLength..
        b2.&maxIPAvailableLength))
-- defined by network operator.
-- Indicates that the resource is available.
-- Its content is network operator specific

IPRoutingAddress {B2: b2}::= CalledPartyNumber {b2}
-- Indicates the routing address for the IP.

IPSSPCapabilities {B2: b2}::= OCTET STRING (SIZE (
                b2.&minIPSSPCapabilitiesLength..
                b2.&maxIPSSPCapabilitiesLength))
-- defined by network operator.
-- Indicates the SRF resources available at the SSP.
-- Its content is network operator specific

ISDNAccessRelatedInformation {B2: b2}::= OCTET STRING (SIZE
        (b2.&minISDNAccessRelatedInfoLength..
         b2.&maxISDNAccessRelatedInfoLength))
-- Indicates the destination user network interface related information. Refer to the ITU-T
Recommendation Q.763 Access
-- Transport parameter for encoding.

LegID::= CHOICE {
    sendingSideID   [0] LegType,
    receivingSideID [1] LegType
    }
-- sendingSideID is used where legID is sent from the SCF  to the SSF and
-- receivingSideID is used  where SCF receives legID from the SSF.

LegType::= OCTET STRING (SIZE(1))

leg1 LegType::= '01'H

leg2 LegType::= '02'H

LocationNumber {B2: b2}::= OCTET STRING (SIZE (
                b2.&minLocationNumberLength..
                b2.&maxLocationNumberLength))
-- Indicates the Location Number for the calling party. Refer to ITU-T Recommendation Q.763
--  for encoding.

MaximumNumberOfCounters::= INTEGER (1..numOfCounters)

MidCallControlInfo {B2: b2}::= SEQUENCE SIZE (
        b2.&minMidCallControlInfoNum..
        b2.&maxMidCallControlInfoNum)
        OF SEQUENCE {
            midCallInfoType         [0] MidCallInfoType {b2},
            midCallReportType       [1] ENUMERATED {
                                        inMonitoringState  (0),
                                        inAnyState         (1)
                                    } DEFAULT inMonitoringState,
            ...
```

```
        }

MidCallInfo {B2: b2}::= SEQUENCE {
    iNServiceControlCode    [0] Digits {b2},
    ...
    }

MidCallInfoType {B2: b2}::= SEQUENCE {
    iNServiceControlCodeLow     [0] Digits {b2},
    iNServiceControlCodeHigh    [1] Digits {b2} OPTIONAL,
    ...
    }

MiscCallInfo::= SEQUENCE {
    messageType [0] ENUMERATED {
                    request         (0),
                    notification    (1)
                    },
        ...
        }
-- Indicates detection point related information.

MonitorMode ::= ENUMERATED {
    interrupted        (0),
    notifyAndContinue  (1),
    transparent        (2)
    }
-- Indicates the event is relayed and/or processed by the SSP.

NAOliInfo        ::= OCTET STRING (SIZE (1))
-- NA Oli information is only used by Camel for NA (North America).
-- It takes the same value as defined in ANSI ISUP T1.113
-- e.g.      '3D'H – Decimal value 61 - Cellular Service (Type 1)
--           '3E'H – Decimal value 62 - Cellular Service (Type 2)
--           '3F'H – Decimal value 63 - Cellular Service (roaming)

NeededCost       ::= SEQUENCE {
    usedCurrencyFactor [1] CurrencyFactor,
    usedScale          [2] CurrencyScale   OPTIONAL,
    reportedCurrency   [3] Currency    DEFAULT noIndication,
    ...
    }

NeededPulses     ::= SEQUENCE {
    usedPulseUnits  [1] OCTET STRING (SIZE (2)),
    ...
    }

noCharge INTEGER::= 0
noScale INTEGER::= 0

normalRelease OCTET STRING::= '839F'H

NumberOfDigits::= INTEGER (1..255)
-- Indicates the number of digits to be collected

OCSIApplicable       ::= NULL
-- Indicates that the Originating CAMEL Subscription Information shall be
-- applied on the outgoing call leg created with a Connect operation. For the use of this
-- parameter see TS 123 078.

OriginalCalledPartyID {B2: b2}::= OCTET STRING (SIZE
            (b2.&minOriginalCalledPartyIDLength..
            b2.&maxOriginalCalledPartyIDLength))
-- Indicates the original called number.
-- Refer to the ITU-T Recommendation Q.763 Original Called Number for encoding.

PartyToCharge   {B2: b2}                ::= SEQUENCE {
    chargedParty                [0] ChargedParty DEFAULT callingPartyNumber,
    specialINNumber             [1] ChargedPartyNumber {b2} OPTIONAL,
    ...
    }
```

```
ProfileIdentifier {B2: b2} ::= CHOICE {
    access      [0] CalledPartyNumber {b2},
    group       [1] FacilityGroup
    }
-- Please note that 'CalledPartyNumber' is used to address a subscriber access line.
-- The data type was reused from the existing types to avoid the definition of a new one.
-- PulseUnits Type


Reason {B2: b2}::= OCTET STRING (SIZE(
    b2.&minReasonLength..
    b2.&maxReasonLength))
-- Its content is network operator specific
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this paramter
-- (after the first tag and length information) is the BER encoding of the defined ASN.1
-- internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.

ReceivingSideID          ::= CHOICE {receivingSideID [1] LegType}
-- used to identify LegID  in operations sent from gsmSSF to gsmSCF

RedirectingPartyID {B2: b2}::= OCTET STRING (SIZE (
            b2.&minRedirectingPartyIDLength..
            b2.&maxRedirectingPartyIDLength))
-- Indicates redirecting number. Refer to the ITU-T Recommendation Q.763 Redirecting number for
encoding.

RedirectionInformation::= OCTET STRING (SIZE(2))
-- Indicates redirection information. Refer to the ITU-T Recommendation Q.763
-- Redirection Information for encoding.

RegistratorIdentifier ::= OCTET STRING
-- Its content is network operator specific

RelayChargesFromDestination        ::= ENUMERATED {
    noIndication          (0),
    relay                 (1),
    noRelay               (2)
    }

ReleaseWhenLimitReached {B2: b2}    ::= SEQUENCE {
    releaseCause       [1] Cause {b2}  DEFAULT normalRelease,
    ...
    }

ReportCondition::= ENUMERATED {
    statusReport   (0),
    timerExpired   (1),
    canceled       (2)
    }
-- ReportCondition specifies the cause of sending "StatusReport" operation to the SCF.

ReportConditions::= SEQUENCE {
    periodicReportinterval     [1] INTEGER (1..32767),
    -- Values are in units of seconds.
    ...
    }

ReportConditionInformation  ::= ENUMERATED {
    intermediateReport         (0),
    finalReportCallReleased    (1),
    finalReportCallActive      (2)
    }

RequestedInformationList {B2: b2}::=
    SEQUENCE SIZE (1..numOfInfoItems) OF RequestedInformation {b2}

RequestedInformationTypeList::= SEQUENCE SIZE (1..numOfInfoItems) OF RequestedInformationType

RequestedInformation {B2: b2}::= SEQUENCE {
    requestedInformationType     [0] RequestedInformationType,
    requestedInformationValue    [1] RequestedInformationValue {b2},
    ...
    }
```

```
RequestedInformationType::= ENUMERATED {
    callAttemptElapsedTime      (0),
    callStopTime                (1),
    callConnectedElapsedTime    (2),
    calledAddress               (3),
    releaseCause                (30)
    }

RequestedInformationValue {B2: b2}::= CHOICE {
    callAttemptElapsedTimeValue     [0] INTEGER (0..255),
    callStopTimeValue               [1] DateAndTime,
    callConnectedElapsedTimeValue   [2] Integer4,
    calledAddressValue              [3] Digits {b2},
    releaseCauseValue               [30] Cause {b2}
    }
-- The callAttemptElapsedTimeValue is specified in seconds.
-- The unit for the callConnectedElapsedTimeValue is 100 milliseconds

RequestedUTSI {B2: b2}::= SEQUENCE {
    uSIServiceIndicator [0] USIServiceIndicator {b2},
    uSImonitorMode      [1] USIMonitorMode,
    ...
    }

RequestedUTSIList {B2: b2}::= SEQUENCE SIZE
            (b2.&minRequestedUTSINum..
            b2.&maxRequestedUTSINum) OF  RequestedUTSI {b2}

ResourceID {B2: b2}::= CHOICE {
    lineID                  [0] Digits {b2},
    facilityGroupID         [1] FacilityGroup,
    facilityGroupMemberID   [2] INTEGER,
    trunkGroupID            [3] INTEGER
    }
-- Indicates a logical identifier for the physical termination resource.

ResourceStatus::= ENUMERATED {
    busy   (0),
    idle   (1)
    }

ResponseCondition::= ENUMERATED {
    intermediateResponse    (0),
    lastResponse            (1)
    }
-- ResponseCondition is used to identify the reason why ServiceFilteringResponse operation is sent.

RouteList {B2: b2}::=
    SEQUENCE SIZE(1..3) OF OCTET STRING (SIZE
            (b2.&minRouteListLength..
             b2.&maxRouteListLength))
-- Indicates a list of trunk groups or a route index..
-- Its content is network operator specific

RouteingNumber  {B2: b2}::= OCTET STRING (SIZE
            (b2.&minRouteingNumberLength..
            b2.&maxRouteingNumberLength))
-- Indicates the Routeing Number.
-- Refer to ITU-T Recommendation Q.763 parameter Network Routeing Number for encoding.


ScfID {B2: b2}::= OCTET STRING (SIZE
                            (b2.&minScfIDLength..
                             b2.&maxScfIDLength))
-- defined by network operator.
-- Indicates the SCF identity.
-- Refer to ITU-T Recommendation Q.713 "calling party address" parameter  for encoding.
-- Other encoding schemes are also possible as a network specific option.

SCIBillingChargingCharacteristics {B2: b2}::= OCTET STRING (SIZE
                            (b2.&minSCIBillingChargingLength..
                             b2.&maxSCIBillingChargingLength))
-- This parameter indicates the billing and/or charging characteristics.
-- Its content is network signalling / operator specific
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this parameter (after the first tag and length
-- information) is the BER encoding of the defined ASN.1 internal structure.
```

```
-- The tag of this parameter as defined by ETSI is never replaced.
-- CAMEL:
--  SCIBillingChargingCharacteristics {PARAMETERS-BOUND: bound}::= OCTET STRING (SIZE (
--  b2.&minSCIBillingChargingLength..b2.&maxSCIBillingChargingLength))
--  (CONSTRAINED BY {
-- shall be the result of the BER-encoded value of type --
--  CAMEL-SCIBillingChargingCharacteristics})
-- Indicates AOC information to be sent to a Mobile Station
-- The violation of the UserDefinedConstraint shall be handled as an ASN.1 syntax error.
-- The value of the SCIBillingChargingCharacteristics of type OCTET STRING carries
-- a value of the ASN.1 data type: CAMEL-SCIBillingChargingCharacteristics.
-- The normal encoding rules are used to encode this value.


SDSSinformation {B2: b2}::= OCTET STRING (SIZE
                           (b2.&minSDSSinformationLength..
                            b2.&maxSDSSinformationLength))

-- Its content is network signalling/operator specific.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this parameter (after the first tag and length
-- information) is the BER encoding of the defined ASN.1 internal structure.

SendingSideID                   ::= CHOICE {sendingSideID  [0] LegType}
-- used to identify LegID in operations sent from gsmSCF to gsmSSF


ServiceInteractionIndicators {B2: b2}::= OCTET STRING (SIZE (
                           b2.&minServiceInteractionIndicatorsLength..
                           b2.&maxServiceInteractionIndicatorsLength))


-- Indicators which are exchanged between SSP and SCP to resolve interactions
-- between IN based services and network based services, respectively
-- between different IN based services.

-- Its content is network signalling/operator specific

-- Note this parameter is kept ifor backward compatibility to IN CS-1,
-- for the present document see parameter ServiceInteractionIndicatorsTwo
-- Its content is network signalling/operator specific.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this parameter (after the first tag and length
-- information) is the BER encoding of the defined ASN.1 internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.

ServiceInteractionIndicatorsTwo::=  SEQUENCE {
-- ServiceInteractionIndicatorsTwo contains Indicators which are exchanged between SSP and SCP
-- to resolve interactions between IN based services  and network based services,
-- respectively between different IN based services.

    forwardServiceInteractionInd       [0] ForwardServiceInteractionInd    OPTIONAL,
-- applicable to operations IDP, CON, ICA, CWA.

    backwardServiceInteractionInd      [1] BackwardServiceInteractionInd   OPTIONAL,
--applicable to operations IDP, CON, CTR, ETC, CWA.

    bothwayThroughConnectionInd        [2] BothwayThroughConnectionInd     OPTIONAL,
-- applicable to operations CTR, ETC.

    suspendTimer                       [3] SuspendTimer    OPTIONAL,
-- applicable to operations CON, ICA CWA.

    connectedNumberTreatmentInd        [4] ConnectedNumberTreatmentInd OPTIONAL,
--applicable to operations CON, CTR, ETC, CWA.

    suppressCallDiversionNotification  [5] BOOLEAN         OPTIONAL,
-- applicable to CON, ICA, CWA

    suppressCallTransferNotification   [6] BOOLEAN         OPTIONAL,
-- applicable to CON, ICA, CWA

    allowCdINNoPresentationInd         [7] BOOLEAN         OPTIONAL,
-- applicable to CON, ICA CWA
-- indicates whether the Address Presentation restricted indicator of the ISUP
-- "called IN number" shall be set to presentation allowed (TRUE)
--  or presentation restricted (FALSE). Refer to ITU-T Recommendation Q.1601.
```

```
    userDialogueDurationInd            [8] BOOLEAN DEFAULT TRUE,
-- applicable to operations CTR, ETC.
-- applicable when interaction with the user is required during call set-up
-- The interaction TRUE means the user interaction may last longer than 90 seconds.
-- Otherwise the indicator should be set to FALSE. Used  for delaying ISUP T9 timer.

    overrideLineRestrictions           [9] BOOLEAN  DEFAULT FALSE,
-- only applicable to operations  (e.g. Connect) which lead to a transition to a PIC before
--  the AuthorizeCallSetup PIC.
-- When set to TRUE, this parameter indicates that some facility restrictions
-- should not be checked when the authority to place a call is verified in the
-- Authorize_Call_Setup PIC.
-- Which restrictions are actually overwriden is network specific.

    suppressVPNAPP                     [10] BOOLEAN DEFAULT FALSE,
-- applicable to CWA, CON, ICA.
-- indicates whether to allow or stop (suppress) the forward transmission of the
-- VPN PSS1 capability.
-- When set to TRUE, the  exchange, on receipt of this parameter, will not transmit for this call
-- any ISUP Application transport parameter with Application Context Identifier set to
-- « PSS1 ASE (VPN) »
-- This indicator  is populated by the SCF, where the SCF and SSF in conjunction have provided the
-- outgoing gateway PINX functionality as required by PSS1.

    calledINNumberOverriding           [11] BOOLEAN    OPTIONAL,
-- applicable to CON and CWA
-- indicates whether the generation/override of  the ISUP
-- "called IN number"  is allowed (TRUE)   or  not allowed (FALSE)
-- If set to FALSE, the ISUP shall not generate a "called IN number"or override
-- an already existing "called IN number".
-- if absent, the default will be "generation/overriding allowed" (TRUE).


-- tag 12 IS RESERVED


    nonCUGCall                         [13] BOOLEAN    DEFAULT FALSE,
-- applicable to CON and CWA
-- indicates whether no parameters for CUG shall be used for the call (i.e. the call shall be
-- a non-CUG call)(TRUE).
-- If set to TRUE, then neither CUG Interlock Code nor Outgoing
-- Access Indicator shall be present; if any of these parameters are present
-- then an error is returned.
-- If set to FALSE or not present, it indicates one of two things:
--  a) continue with modified CUG information (when one or more of either CUG Interlock Code
--     and Outgoing Access Indicator are present), or
--  b) continue with original CUG information (when neither CUG Interlock Code nor Outgoing
--      Access Indicator are present).
...
}


ServiceKey::= Integer4
-- Information that allows the SCF to choose the appropriate service logic.


SFBillingChargingCharacteristics {B2: b2}::= OCTET STRING (SIZE
                            (b2.&minSFBillingChargingLength..
                             b2.&maxSFBillingChargingLength))
-- This parameter indicates the billing and/or charging characteristics for filtered calls.
-- Its content is network signalling/operator specific.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this parameter (after the first tag and length
-- information) is the BER encoding of the defined ASN.1 internal structure.
-- The tag of this parameter as defined by ETSI is never replaced

SSFdetermination::= ENUMERATED{
      noDetermination             (0),
      destinationRoutingAddress   (1),
      calledInNumber              (2)
      }
-- The calledInNumber is the calledPartyNumber in the InitialDP operation.


-- SubTariffControl Type
```

```
SupervisionMethod    ::= CHOICE {
    maximumTariffCurrency   [1] SEQUENCE {
        currencyFactor          [0] INTEGER (1..999999),
        -- Values are in currency units.
        currencyScale           [1] CurrencyScale DEFAULT noScale,
        -- Imported from ES 201 296.
        currency                [2] Currency    DEFAULT noIndication,
        -- Imported from ES 201 296.
        ...
        },
    maximumTimeDuration     [2] SEQUENCE {
        maximumDurationAllowed  [0] INTEGER (1..864000),
        -- Values are in units of 100 msec.
        switchOverTime          [1] INTEGER (1..86400)  OPTIONAL,
        -- Values are in units of seconds.
        ...
        },
    maximumPulseUnits       [3] OCTET STRING (SIZE (2))
    }

SupportedTriggers   TRIGGER::= {...}

SuspendTimer::= INTEGER (-1..120)
-- value in seconds
-- The default is as specified in EN 301 070-1
-- The value -1 indicates the network specific suspend timer (T6) is to be used.


TariffFromSuccExchange          ::= ENUMERATED {
    notTakenIntoAccount             (0),
    takeIntoAccountNoAOC            (1),
    takeIntoAccountTranslateIntoAOC (2)
    }

TariffInfoFromSuccExchangeSCI   ::= SEQUENCE {
    relayChargesFromDestination     [0] RelayChargesFromDestination    DEFAULT noRelay,
    relayAOCfromDestination         [1] BOOLEAN    DEFAULT FALSE,
    ...
    }

TDPIdentifier {B2: b2} ::=  CHOICE {

    oneTrigger      INTEGER,
    triggers    [1] Triggers {b2}
    }

TerminalType::= ENUMERATED {
    unknown     (0),
    dialPulse   (1),
    dtmf        (2),
    isdn        (3),
    isdnNoDtmf  (4),
    spare       (16)
    }

TimeAndTimezone {B2: b2}::= OCTET STRING (SIZE(b2.&minTimeAndTimezoneLength..
                                               b2.&maxTimeAndTimezoneLength))
-- Support for CAMEL
-- Indicates the time and timezone, relative to GMT. This parameter BCD encoded.
-- The year digit indicating millenium occupies bits 0-3 of the first octet, and the year
-- digit indicating century occupies bits 4-7 of the first octet.
-- The year digit indicating decade occupies bits 0-3 of the second octet, whilst the digit
-- indicating the year within the decade occupies bits 4-7 of the second octet.
-- The most significant month digit occupies bits 0-3 of the third octet, and the least
-- significant month digit occupies bits 4-7 of the third octet.
-- The most significant day digit occupies bits 0-3 of the fourth octet, and the least
-- significant day digit occupies bits 4-7 of the fourth octet.
-- The most significant hours digit occupies bits 0-3 of the fifth octet, and the least
-- significant hours digit occupies bits 4-7 of the fifth octet.
-- The most significant minutes digit occupies bits 0-3 of the sixth octet, and the least
-- significant minutes digit occupies bits 4-7 of the sixth octet.
-- The most significant seconds digit occupies bits 0-3 of the seventh octet, and the least
-- significant seconds digit occupies bits 4-7 of the seventh octet.
--
-- The timezone information occupies the eigth octet. For the encoding of Timezone refer to
-- Reference [29], TS 123 040.
--
```

```
--  The BCD digits are packed and encoded as follows:
--
--  Bit  7  6  5   4  |  3  2  1  0
--       2nd digit    | 1st digit         Octet 1
--       3rd digit    | 4th digit         Octet 2
--                    ..                   ..
--       nth digit    | n-1th digit       Octet m
--
--       0000     digit 0
--       0001     digit 1
--       0010     digit 2
--       0011     digit 3
--       0100     digit 4
--       0101     digit 5
--       0110     digit 6
--       0111     digit 7
--       1000     digit 8
--       1001     digit 9
--       1010     spare
--       1011     spare
--       1100     spare
--       1101     spare
--       1110     spare
--       1101     spare
--
-- where the leftmost bit of the digit is either bit 7 or bit 3 of the octet.

TimeInformation                     ::= CHOICE {
    timeIfNoTariffSwitch                [0] TimeIfNoTariffSwitch,
    timeIfTariffSwitch                  [1] TimeIfTariffSwitch
    }
-- Indicates call duration information
-- Support for CAMEL

TimeIfNoTariffSwitch                ::= INTEGER(0..864000)
-- TimeIfNoTariffSwitch is measured in 100 millisecond intervals
-- Support for CAMEL

TimeIfTariffSwitch                  ::= SEQUENCE {
    timeSinceTariffSwitch               [0] INTEGER(0..864000),
    tariffSwitchInterval                [1] INTEGER(1..864000)        OPTIONAL,
    ...
    }
-- timeSinceTariffSwitch and tariffSwitchInterval are measured in 100 millisecond intervals
-- Support for CAMEL

TimerID::= ENUMERATED {
    tssf(0)
    }
-- Indicates the timer to be reset.

TimerValue::= Integer4
-- Indicates the timer value (in seconds).

Tone ::= SEQUENCE {
    toneID      [0] Integer4,
    duration    [1] Integer4          OPTIONAL,
    ...
    }
-- The duration specifies the length of the tone in seconds, value 0 indicates infinite duration.

TriggerData::= SEQUENCE {
    triggerId       [0]     TRIGGER.&id      ({SupportedTriggers}),
    triggerPar      [1]     TRIGGER.&Parameter ({SupportedTriggers}{@triggerId}),
    ...
    }

TriggerDataIdentifier {B1: b1, B2: b2}::= SEQUENCE {
    triggerID       [0] EventTypeBCSM,
    profile         [1] ProfileIdentifier {b2},
    extensions      [2] Extensions  {b1}    OPTIONAL,
    ...
    }
-- It is outside the scope of this capability set  whether all TDP types really apply
```

```
TriggerDPType::= ENUMERATED {tdp-r(0)}

TriggerResults {B2: b2}::=
    SEQUENCE SIZE (1..b2.&numOfTriggers) OF TriggerResult

TriggerResult ::=SEQUENCE {
    tDPIdentifer       [0] INTEGER,
    actionPerformed    [1] ActionPerformed,
    dPName             [2] EventTypeBCSM OPTIONAL,
    ...
    }

Triggers {B2: b2}::=SEQUENCE SIZE(1..b2.&numOfTriggers) OF Trigger

Trigger ::=SEQUENCE {
    tDPIdentifier      [0] INTEGER,
    dpName             [1] EventTypeBCSM OPTIONAL,
    ...
    }

TriggerStatus::= ENUMERATED {
            created         (0),
            alreadyExist    (1),
            deleted         (2),
            unknownTrigger  (3)
            }

UsedCost        ::= SEQUENCE {
    usedCurrencyFactor [1] CurrencyFactor,
-- imported from ES 201 296
    usedScale          [2] CurrencyScale   OPTIONAL,
-- imported from ES 201 296
    reportedCurrency   [3] Currency     DEFAULT noIndication,
-- imported from ES 201 296
    ...
    }

UsedTime    ::= SEQUENCE {
    usedDuration                   [1] INTEGER (0..864000),
    -- Units are in 100 msec.
    usedDurationAfterSwitchOverTime [2] INTEGER (0..864000) OPTIONAL,
    -- Units are in 100 msec.
    ...
    }

UsedPulses  ::= SEQUENCE {
    usedPulseUnits [1] OCTET STRING (SIZE (2)),
    ...
    }

USIInformation {B2: b2}::= OCTET STRING (SIZE
            (b2.&minUSIInformationLength..
             b2.&maxUSIInformationLength))
-- Its content is network signalling/operator specific
-- Indicates the length of the USIInformation element, maxUSIInformationlength will depend on
-- the constraints imposed by the network signalling used to transport the USI information.
-- Its content is network signalling/operator specific.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this parameter (after the first tag and length
-- information) is the BER encoding of the defined ASN.1 internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.

USIMonitorMode::= ENUMERATED {
    monitoringActive    (0),
    monitoringInactive  (1)
    }

-- Indicates if the relationship for the specified UTSI IE should be activated or deactivated.

USIServiceIndicator {B2: b2}::= CHOICE  {
    global  OBJECT IDENTIFIER,
    local   OCTET STRING (SIZE (
            b2.&minUSIServiceIndicatorLength..
            b2.&maxUSIServiceIndicatorLength))
```

```
}
-- In case of local its content is network signalling/operator specific
-- Its content is network signalling/operator specific.
-- The internal structure of this parameter can be defined using ASN.1 and the related Basic
-- Encoding Rules (BER). In such a case the value of this parameter (after the first tag and length
-- information) is the BER encoding of the defined ASN.1 internal structure.
-- The tag of this parameter as defined by ETSI is never replaced.

VPNIndicator::= BOOLEAN
-- This parameter is set to TRUE if the originating call part supports
-- VPN with PSS1 information flows

WarningBeforeLimitReached   ::= SEQUENCE {
    durationBeforeLimitReached [1] INTEGER (1..255),
    -- The values are in units of 1 second.
    warningToSend              [2] Tone        OPTIONAL,
    warningDirection           [3] LegID          OPTIONAL,
    ...
    }

-- The Definition of range of constants Follows
highLayerCompatibilityLength    INTEGER::= 2
minCauseLength                  INTEGER::= 2
numOfCounters                   INTEGER::= 100
numOfInfoItems                  INTEGER::= 5

END
```

14.2   Classes

```
IN-CS3-SSF-SCF-Classes {itu-t(0) identified-organization(4) etsi(0) inDomain(1) in-network(1)
cs3(30) modules(1) in-cs3-ssf-scf-classes(7) version1(0)}


DEFINITIONS::=

BEGIN



TRIGGER::= CLASS {
    &Parameter  OPTIONAL,
    &id        INTEGER UNIQUE
    }
WITH SYNTAX {
    [PARAMETER      &Parameter]
    IDENTIFIED BY   &id
    }

--The &id field uniquely identifies a class of triggers.
--The &Parameter field defines the ASN.1 type for representing specific parameters
-- (e.g. criteria, scfAddress,...)
 --associated with this class of triggers.
SCF-SSF-BOUNDS::= CLASS {
    &minAChBillingChargingLength                INTEGER   OPTIONAL,
    &maxAChBillingChargingLength                INTEGER   OPTIONAL,
    &minBackwardGVNSLength                      INTEGER   OPTIONAL,
    &maxBackwardGVNSLength                      INTEGER   OPTIONAL,
    &maxBearerCapabilityLength                  INTEGER   OPTIONAL,
    &minCalledDirectoryNumberLength             INTEGER   OPTIONAL,
    &maxCalledDirectoryNumberLength             INTEGER   OPTIONAL,
    &minCalledPartyBCDNumberLength              INTEGER   OPTIONAL,
    &maxCalledPartyBCDNumberLength              INTEGER   OPTIONAL,
    &minCalledPartyNumberLength                 INTEGER   OPTIONAL,
    &maxCalledPartyNumberLength                 INTEGER   OPTIONAL,
    &minCallingGeodeticLocationLength           INTEGER   OPTIONAL,
    &maxCallingGeodeticLocationLength           INTEGER   OPTIONAL,
    &minCallingPartyNumberLength                INTEGER   OPTIONAL,
    &maxCallingPartyNumberLength                INTEGER   OPTIONAL,
    &minCallResultcs1Length                     INTEGER   OPTIONAL,
    &maxCallResultcs1Length                     INTEGER   OPTIONAL,
    &maxCallReferenceLength                     INTEGER   OPTIONAL,
    &minCarrierLength                           INTEGER   OPTIONAL,
    &maxCarrierLength                           INTEGER   OPTIONAL,
    &minChargedPartyNumberLength                INTEGER   OPTIONAL,
```

```
&maxChargedPartyNumberLength                     INTEGER     OPTIONAL,
&maxCauseLength                                  INTEGER     OPTIONAL,
&minCommunicationTariffNum                       INTEGER     OPTIONAL,
&maxCommunicationTariffNum                       INTEGER     OPTIONAL,
&maxCNInfoLength                                 INTEGER     OPTIONAL,
&minDigitsLength                                 INTEGER     OPTIONAL,
&maxDigitsLength                                 INTEGER     OPTIONAL,
&minDisplayInformationLength                     INTEGER     OPTIONAL,
&maxDisplayInformationLength                     INTEGER     OPTIONAL,
&minEventSpecificInformationChargingLength       INTEGER     OPTIONAL,
&maxEventSpecificInformationChargingLength       INTEGER     OPTIONAL,
&minEventTypeChargingLength                      INTEGER     OPTIONAL,
&maxEventTypeChargingLength                      INTEGER     OPTIONAL,
&minFCIBCCcs1Length                              INTEGER     OPTIONAL,
&maxFCIBCCcs1Length                              INTEGER     OPTIONAL,
&minFCIBCCcs2Length                              INTEGER     OPTIONAL,
&maxFCIBCCcs2Length                              INTEGER     OPTIONAL,
&minFCIBillingChargingDataLength                 INTEGER     OPTIONAL,
&maxFCIBillingChargingDataLength                 INTEGER     OPTIONAL,
&minCamelFCIBillingChargingDataLength            INTEGER     OPTIONAL,
&maxCamelFCIBillingChargingDataLength            INTEGER     OPTIONAL,
&minFCIBillingChargingLength                     INTEGER     OPTIONAL,
&maxFCIBillingChargingLength                     INTEGER     OPTIONAL,
&minFormatDataLength                             INTEGER     OPTIONAL,
&maxFormatDataLength                             INTEGER     OPTIONAL,
&minForwardGVNSLength                            INTEGER     OPTIONAL,
&maxForwardGVNSLength                            INTEGER     OPTIONAL,
&minGenericNameLength                            INTEGER     OPTIONAL,
&maxGenericNameLength                            INTEGER     OPTIONAL,
&minGenericNumberLength                          INTEGER     OPTIONAL,
&maxGenericNumberLength                          INTEGER     OPTIONAL,
&maxGlobalCallReferenceLength                    INTEGER     OPTIONAL,
&maxInitialTimeInterval                          INTEGER     OPTIONAL,
&maxINServiceCompatibilityIndLength              INTEGER     OPTIONAL,
&minIPAvailableLength                            INTEGER     OPTIONAL,
&maxIPAvailableLength                            INTEGER     OPTIONAL,
&minIPSSPCapabilitiesLength                      INTEGER     OPTIONAL,
&maxIPSSPCapabilitiesLength                      INTEGER     OPTIONAL,
&minISDNAccessRelatedInfoLength                  INTEGER     OPTIONAL,
&maxISDNAccessRelatedInfoLength                  INTEGER     OPTIONAL,
&minLocationNumberLength                         INTEGER     OPTIONAL,
&maxLocationNumberLength                         INTEGER     OPTIONAL,
&minMidCallControlInfoNum                        INTEGER     OPTIONAL,
&maxMidCallControlInfoNum                        INTEGER     OPTIONAL,
&minOriginalCalledPartyIDLength                  INTEGER     OPTIONAL,
&maxOriginalCalledPartyIDLength                  INTEGER     OPTIONAL,
&minReasonLength                                 INTEGER     OPTIONAL,
&maxReasonLength                                 INTEGER     OPTIONAL,
&minRedirectingPartyIDLength                     INTEGER     OPTIONAL,
&maxRedirectingPartyIDLength                     INTEGER     OPTIONAL,
&minRequestedUTSINum                             INTEGER     OPTIONAL,
&maxRequestedUTSINum                             INTEGER     OPTIONAL,
&minRouteListLength                              INTEGER     OPTIONAL,
&maxRouteListLength                              INTEGER     OPTIONAL,
&minRouteingNumberLength                         INTEGER     OPTIONAL,
&maxRouteingNumberLength                         INTEGER     OPTIONAL,
&minScfIDLength                                  INTEGER     OPTIONAL,
&maxScfIDLength                                  INTEGER     OPTIONAL,
&minSCIBillingChargingLength                     INTEGER     OPTIONAL,
&maxSCIBillingChargingLength                     INTEGER     OPTIONAL,
&minSDSSinformationLength                        INTEGER     OPTIONAL,
&maxSDSSinformationLength                        INTEGER     OPTIONAL,
&minServiceInteractionIndicatorsLength           INTEGER     OPTIONAL,
&maxServiceInteractionIndicatorsLength           INTEGER     OPTIONAL,
&minSFBillingChargingLength                      INTEGER     OPTIONAL,
&maxSFBillingChargingLength                      INTEGER     OPTIONAL,
&minSubTariffControlLen                          INTEGER     OPTIONAL,
&maxSubTariffControlLen                          INTEGER     OPTIONAL,
&minTimeAndTimezoneLength                        INTEGER     OPTIONAL,
&maxTimeAndTimezoneLength                        INTEGER     OPTIONAL,
&minTariffIndicatorsLen                          INTEGER     OPTIONAL,
&maxTariffIndicatorsLen                          INTEGER     OPTIONAL,
&minUSIInformationLength                         INTEGER     OPTIONAL,
&maxUSIInformationLength                         INTEGER     OPTIONAL,
&minUSIServiceIndicatorLength                    INTEGER     OPTIONAL,
&maxUSIServiceIndicatorLength                    INTEGER     OPTIONAL,
&numOfBCSMEvents                                 INTEGER     OPTIONAL,
&numOfBCUSMEvents                                INTEGER     OPTIONAL,
```

```
       &numOfChargingEvents                           INTEGER    OPTIONAL,
       &numOfCSAs                                      INTEGER    OPTIONAL,
       &numOfCSs                                       INTEGER    OPTIONAL,
       &numOfGenericNumbers                            INTEGER    OPTIONAL,
       &numOfINProfile                                 INTEGER    OPTIONAL,
       &numOfTriggers                                  INTEGER    OPTIONAL,
       &numOfInServiceCompatibilityIndLength           INTEGER    OPTIONAL,
       &numOfLegs                                      INTEGER    OPTIONAL,
       &numOfMessageIDs                                INTEGER    OPTIONAL,
       &maxAmount                                      INTEGER    OPTIONAL,
       &maxInitialUnitIncrement                        INTEGER    OPTIONAL,
       &maxScalingFactor                               INTEGER    OPTIONAL,
       &maxSegmentsPerDataInterval                     INTEGER    OPTIONAL,
       &ub-nbCall                                      INTEGER    OPTIONAL,
       &numOfAddresses                                 INTEGER    OPTIONAL


     }
WITH SYNTAX
     {
     [MINIMUM-FOR-ACH-BILLING-CHARGING               &minAChBillingChargingLength]
     [MAXIMUM-FOR-ACH-BILLING-CHARGING               &maxAChBillingChargingLength]
     [MINIMUM-FOR-BACKWARD-GVNS                       &minBackwardGVNSLength]
     [MAXIMUM-FOR-BACKWARD-GVNS                       &maxBackwardGVNSLength]
     [MAXIMUM-FOR-BEARER-CAPABILITY                   &maxBearerCapabilityLength]
     [MINIMUM-FOR-CALLED-DIRECTORY-NUMBER             &minCalledDirectoryNumberLength]
     [MAXIMUM-FOR-CALLED-DIRECTORY-NUMBER             &maxCalledDirectoryNumberLength]
     [MINIMUM-FOR-CALLED-PARTY-BCD-NUMBER             &minCalledPartyBCDNumberLength]
     [MAXIMUM-FOR-CALLED-PARTY-BCD-NUMBER             &maxCalledPartyBCDNumberLength]
     [MINIMUM-FOR-CALLED-PARTY-NUMBER                 &minCalledPartyNumberLength]
     [MAXIMUM-FOR-CALLED-PARTY-NUMBER                 &maxCalledPartyNumberLength]
     [MINIMUM-FOR-CALLING-GEODETIC-LOCATION           &minCallingGeodeticLocationLength]
     [MAXIMUM-FOR-CALLING-GEODETIC-LOCATION           &maxCallingGeodeticLocationLength]
     [MINIMUM-FOR-CALLING-PARTY-NUMBER                &minCallingPartyNumberLength]
     [MAXIMUM-FOR-CALLING-PARTY-NUMBER                &maxCallingPartyNumberLength]
     [MINIMUM-FOR-CALL-RESULT-CSONE                   &minCallResultcs1Length]
     [MAXIMUM-FOR-CALL-RESULT-CSONE                   &maxCallResultcs1Length]
     [MAXIMUM-FOR-CALL-REFERENCE                      &maxCallReferenceLength]
     [MINIMUM-FOR-CARRIER                             &minCarrierLength]
     [MAXIMUM-FOR-CARRIER                             &maxCarrierLength]
     [MINIMUM-FOR-CHARGED-PARTYNUMBER                 &minChargedPartyNumberLength]
     [MAXIMUM-FOR-CHARGED-PARTYNUMBER                 &maxChargedPartyNumberLength]
     [MAXIMUM-FOR-CAUSE                               &maxCauseLength]
     [MINIMUM-FOR-COMMUNICATION-TARIFF-NUM            &minCommunicationTariffNum]
     [MAXIMUM-FOR-COMMUNICATION-TARIFF-NUM            &maxCommunicationTariffNum]
     [MAXIMUM-FOR-CNINFO                              &maxCNInfoLength]
     [MINIMUM-FOR-DIGITS                              &minDigitsLength]
     [MAXIMUM-FOR-DIGITS                              &maxDigitsLength]
     [MINIMUM-FOR-DISPLAY                             &minDisplayInformationLength]
     [MAXIMUM-FOR-DISPLAY                             &maxDisplayInformationLength]
     [MINIMUM-FOR-EVENT-SPECIFIC-CHARGING             &minEventSpecificInformationChargingLength]
     [MAXIMUM-FOR-EVENT-SPECIFIC-CHARGING             &maxEventSpecificInformationChargingLength]
     [MINIMUM-FOR-EVENT-TYPE-CHARGING                 &minEventTypeChargingLength]
     [MAXIMUM-FOR-EVENT-TYPE-CHARGING                 &maxEventTypeChargingLength]
     [MINIMUM-FOR-FCI-CSONE-BILLING-CHARGING          &minFCIBCCcs1Length]
     [MAXIMUM-FOR-FCI-CSONE-BILLING-CHARGING          &maxFCIBCCcs1Length]
     [MINIMUM-FOR-FCI-CSTWO-BILLING-CHARGING          &minFCIBCCcs2Length]
     [MAXIMUM-FOR-FCI-CSTWO-BILLING-CHARGING          &maxFCIBCCcs2Length]
     [MINIMUM-FOR-FCI-BILLING-CHARGING-DATA           &minFCIBillingChargingDataLength]
     [MAXIMUM-FOR-FCI-BILLING-CHARGING-DATA           &maxFCIBillingChargingDataLength]
     [MINIMUM-FOR-CAMELFCI-BILLING-CHARGING-DATA &minCamelFCIBillingChargingDataLength]
     [MAXIMUM-FOR-CAMELFCI-BILLING-CHARGING-DATA &maxCamelFCIBillingChargingDataLength]
     [MINIMUM-FOR-FCI-BILLING-CHARGING                &minFCIBillingChargingLength]
     [MAXIMUM-FOR-FCI-BILLING-CHARGING                &maxFCIBillingChargingLength]
     [MINIMUM-FOR-FORMAT-DATA-LENGTH                  &minFormatDataLength]
     [MAXIMUM-FOR-FORMAT-DATA-LENGTH                  &maxFormatDataLength]
     [MINIMUM-FOR-FORWARD-GVNS                        &minForwardGVNSLength]
     [MAXIMUM-FOR-FORWARD-GVNS                        &maxForwardGVNSLength]
     [MINIMUM-FOR-GENERIC-NAME                        &minGenericNameLength]
     [MAXIMUM-FOR-GENERIC-NAME                        &maxGenericNameLength]
     [MINIMUM-FOR-GENERIC-NUMBER                      &minGenericNumberLength]
     [MAXIMUM-FOR-GENERIC-NUMBER                      &maxGenericNumberLength]
     [MAXIMUM-FOR-GLOBAL-CALLREF                      &maxGlobalCallReferenceLength]
     [MAXIMUM-FOR-INITIAL-TIME-INTERVAL               &maxInitialTimeInterval]
     [MAXIMUM-FOR-IN-SERVICE-COMPATIBILITY            &maxINServiceCompatibilityIndLength]
     [MINIMUM-FOR-IP-AVAILABLE                        &minIPAvailableLength]
     [MAXIMUM-FOR-IP-AVAILABLE                        &maxIPAvailableLength]
     [MINIMUM-FOR-IP-SSP-CAPABILITIES                 &minIPSSPCapabilitiesLength]
     [MAXIMUM-FOR-IP-SSP-CAPABILITIES                 &maxIPSSPCapabilitiesLength]
```

```
    [MINIMUM-FOR-ISDN-ACCESS-RELATED-INFO          &minISDNAccessRelatedInfoLength]
    [MAXIMUM-FOR-ISDN-ACCESS-RELATED-INFO          &maxISDNAccessRelatedInfoLength]
    [MINIMUM-FOR-LOCATION-NUMBER                   &minLocationNumberLength]
    [MAXIMUM-FOR-LOCATION-NUMBER                   &maxLocationNumberLength]
    [MINIMUM-FOR-MID-CALL-CONTROL-INFO             &minMidCallControlInfoNum]
    [MAXIMUM-FOR-MID-CALL-CONTROL-INFO             &maxMidCallControlInfoNum]
    [MINIMUM-FOR-ORIGINAL-CALLED-PARTY-ID          &minOriginalCalledPartyIDLength]
    [MAXIMUM-FOR-ORIGINAL-CALLED-PARTY-ID          &maxOriginalCalledPartyIDLength]
    [MINIMUM-FOR-REASON                            &minReasonLength]
    [MAXIMUM-FOR-REASON                            &maxReasonLength]
    [MINIMUM-FOR-REDIRECTING-ID                    &minRedirectingPartyIDLength]
    [MAXIMUM-FOR-REDIRECTING-ID                    &maxRedirectingPartyIDLength]
    [MINIMUM-FOR-REQUESTED-UTSI-NUM                &minRequestedUTSINum]
    [MAXIMUM-FOR-REQUESTED-UTSI-NUM                &maxRequestedUTSINum]
    [MINIMUM-FOR-ROUTE-LIST                        &minRouteListLength]
    [MAXIMUM-FOR-ROUTE-LIST                        &maxRouteListLength]
    [MINIMUM-FOR-ROUTING-NUMBER                    &minRouteingNumberLength]
    [MAXIMUM-FOR-ROUTING-NUMBER                    &maxRouteingNumberLength]
    [MINIMUM-FOR-SCF-ID                            &minScfIDLength]
    [MAXIMUM-FOR-SCF-ID                            &maxScfIDLength]
    [MINIMUM-FOR-SCI-BILLING-CHARGING              &minSCIBillingChargingLength]
    [MAXIMUM-FOR-SCI-BILLING-CHARGING              &maxSCIBillingChargingLength]
    [MINIMUM-FOR-SDSS-INFORMATION                  &minSDSSinformationLength]
    [MAXIMUM-FOR-SDSS-INFORMATION                  &maxSDSSinformationLength]
    [MINIMUM-FOR-SII                               &minServiceInteractionIndicatorsLength]
    [MAXIMUM-FOR-SII                               &maxServiceInteractionIndicatorsLength]
    [MINIMUM-FOR-SF-BILLING-CHARGING               &minSFBillingChargingLength]
    [MAXIMUM-FOR-SF-BILLING-CHARGING               &maxSFBillingChargingLength]
    [MINIMUM-FOR-SUB-TARRIF-CONTROL-LENGTH         &minSubTariffControlLen]
    [MAXIMUM-FOR-SUB-TARRIF-CONTROL-LENGTH         &maxSubTariffControlLen]
    [MINIMUM-FOR-TIME-AND-TIMEZONE-LENGTH          &minTimeAndTimezoneLength]
    [MAXIMUM-FOR-TIME-AND-TIMEZONE-LENGTH          &maxTimeAndTimezoneLength]
    [MINIMUM-FOR-TARIFF-INDICATORS-LENGTH          &minTariffIndicatorsLen]
    [MAXIMUM-FOR-TARIFF-INDICATORS-LENGTH          &maxTariffIndicatorsLen]
    [MINIMUM-FOR-USI-INFORMATION                   &minUSIInformationLength]
    [MAXIMUM-FOR-USI-INFORMATION                   &maxUSIInformationLength]
    [MINIMUM-FOR-USI-SERVICE-INDICATOR             &minUSIServiceIndicatorLength]
    [MAXIMUM-FOR-USI-SERVICE-INDICATOR             &maxUSIServiceIndicatorLength]
    [NUM-OF-BCSM-EVENT                             &numOfBCSMEvents]
    [NUM-OF-BCUSM-EVENT                            &numOfBCUSMEvents]
    [NUM-OF-CHARGING-EVENT                         &numOfChargingEvents]
    [NUM-OF-CSAS                                   &numOfCSAs]
    [NUM-OF-CSS                                    &numOfCSs]
    [NUM-OF-GENERIC-NUMBERS                        &numOfGenericNumbers]
    [NUM-OF-INPROFILE                              &numOfINProfile]
    [NUM-OF-SEVERALTRIGGER                         &numOfTriggers]
    [NUM-OF-IN-SERVICE-COMPATIBILITY-ID            &numOfInServiceCompatibilityIndLength]
    [NUM-OF-LEGS                                   &numOfLegs]
    [NUM-OF-MESSAGE-IDS                            &numOfMessageIDs]
    [MAXIMUM-FOR-AMOUNT                            &maxAmount]
    [MAXIMUM-FOR-INITIAL-UNIT-INCREMENT            &maxInitialUnitIncrement]
    [MAXIMUM-FOR-SCALING-FACTOR                    &maxScalingFactor]
    [MAXIMUM-FOR-SEGMENTS-PER-DATA-INTERVAL        &maxSegmentsPerDataInterval]
    [MAXIMUM-FOR-UB-NB-CALL                        &ub-nbCall]
    [NUM-OF-ADDRESSES                             &numOfAddresses]

    }


END
```

14.3 Operations and Arguments

```
IN-CS3-SSF-SCF-ops-args {itu-t(0) identified-organization(4) etsi(0) inDomain(1) in-network(1)
cs3(30) modules(1) in-cs3-ssf-scf-ops-args(8) version1(0)}


DEFINITIONS IMPLICIT TAGS::=

BEGIN

IMPORTS

    common-classes,
    common-datatypes,
    errortypes,
    scf-srf-classes,
    scf-srf-datatypes,
    ssf-scf-classes,
    ssf-scf-datatypes,
    operationcodes,
    ros-InformationObjects,
    tc-Messages
FROM IN-CS3-object-identifiers
    {itu-t(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) cs3(30) modules(1) in-
cs3-object-identifiers(0) version1(0)}

    OPERATION

FROM Remote-Operations-Information-Objects ros-InformationObjects


    AddOnChargingInformation,
    ChargingTariffInformation,
    ChargingMessageType
FROM Tariffing-DataTypes {itu-t(0) identified-organization etsi (0) 1296  version2(3)}

    ISDN-AddressString,
    IMSI,
    Ext-BasicServiceCode
-- Refer to TS 129 002 for encoding.
FROM MAP-CommonDataTypes {ITU-T(0) identified-organization(4) etsi(0) mobileDomain(0)
gsm-Network(1) modules(3) map-CommonDataTypes(18) version6(6)}

    CUG-Index,
    CUG-Interlock,
    LocationInformation,
-- Refer to TS 129 002 for encoding.
    SubscriberState
FROM MAP-MS-DataTypes {ITU-T(0) identified-organization(4) etsi(0) mobileDomain(0)
gsm-Network(1) modules(3) map-MS-DataTypes(11) version6(6)}



    CallReferenceNumber,
-- Refer to TS 129 002 for encoding.
    SuppressionOfAnnouncement
FROM MAP-CH-DataTypes {ITU-T(0) identified-organization(4) etsi(0) mobileDomain(0)
gsm-Network(1) modules(3) map-CH-DataTypes(13) version6(6)}


    COMMON-BOUNDS
FROM IN-CS3-common-classes common-classes

    SCF-SSF-BOUNDS
FROM IN-CS3-SSF-SCF-Classes ssf-scf-classes

    SCF-SRF-BOUNDS
FROM IN-CS3-scf-srf-classes scf-srf-classes

    opcode-activateServiceFiltering,
    opcode-activityTest,
    opcode-applyCharging,
    opcode-applyChargingReport,
    opcode-assistRequestInstructions,
    opcode-callGap,
```

```
        opcode-callInformationReport,
        opcode-callInformationRequest,
        opcode-cancel,
        opcode-cancelStatusReportRequest,
        opcode-collectInformation,
        opcode-connect,
        opcode-connectToResource,
        opcode-continue,
        opcode-continueWithArgument,
        opcode-createCallSegmentAssociation,
        opcode-createOrRemoveTriggerData,
        opcode-disconnectForwardConnection,
        opcode-dFCWithArgument,
        opcode-disconnectLeg,
        opcode-entityReleased,
        opcode-establishTemporaryConnection,
        opcode-eventNotificationCharging,
        opcode-eventReportBCSM,
        opcode-furnishChargingInformation,
        opcode-initialDP,
        opcode-initiateCallAttempt,
        opcode-manageTriggerData,
        opcode-mergeCallSegments,
        opcode-moveCallSegments,
        opcode-moveLeg,
        opcode-releaseCall,
        opcode-reportUTSI,
        opcode-requestCurrentStatusReport,
        opcode-requestEveryStatusChangeReport,
        opcode-requestFirstStatusMatchReport,
        opcode-requestNotificationChargingEvent,
        opcode-requestReportBCSMEvent,
        opcode-requestReportUTSI,
        opcode-resetTimer,
        opcode-selectFacility,
        opcode-sendChargingInformation,
        opcode-sendSTUI,
        opcode-serviceFilteringResponse,
        opcode-setServiceProfile,
        opcode-splitLeg,
        opcode-statusReport

FROM IN-CS3-operationcodes operationcodes

        Extensions{},
        Integer4,
        InvokeID

FROM IN-CS3-common-datatypes common-datatypes


                ActionIndicator,
                ActionPerformed,
                AChBillingChargingCharacteristics {},
                AChChargingAddress {},
                AdditionalCallingPartyNumber {},
                AlertingPattern,
                ApplicationTimer,
                AssistingSSPIPRoutingAddress {},
                BackwardGVNS {},
                BCSMEvent  {},
                BearerCapability {},
                CalledDirectoryNumber {},
                CalledPartyNumber {},
                CalledPartyBCDNumber {},
                CallingGeodeticLocation {},
                CallingPartyBusinessGroupID,
                CallingPartyNumber {},
                CallingPartysCategory,
                CallReference {},
                CallResult {},
                CallSegmentID {},
                CallSupervision {},
                Carrier {},
                Cause {},
                CCSS,
                CGEncountered,
                ChargeNumber {},
```

```
ChargingControlType,
ChargingEvent {},
CNInfo {},
ControlType,
CorrelationID {},
CountersValue,
CreateOrRemoveIndicator,
CSAID {},
CutAndPaste,
DateAndTime,
DefaultFaultHandling {},
DestinationRoutingAddress {},
Digits {},
DisplayInformation {},
Duration,
EventSpecificInformationBCSM {},
EventSpecificInformationCharging {},
EventTypeBCSM,
EventTypeCharging {},
EventTypeTariff,
FCIBillingChargingCharacteristics {},
FilteredCallTreatment {},
FilteringCharacteristics,
FilteringCriteria {},
FilteringTimeOut,
ForwardCallIndicators,
ForwardGVNS {},
GapCriteria {},
GapIndicators,
GapTreatment {},
GenericName {},
GenericNumbers {},
GlobalCallReference {},
HighLayerCompatibility,
initialCallSegment,
InitialDPArgExtension,
INprofile {},
INServiceCompatibilityIndication {},
INServiceCompatibilityResponse,
IPAvailable {},
IPRoutingAddress {},
IPSSPCapabilities {},
ISDNAccessRelatedInformation {},
LegID,
leg1,
LocationNumber {},
MiscCallInfo,
MonitorMode,
NAOliInfo,
OCSIApplicable,
OriginalCalledPartyID {},
ProfileIdentifier {},
Reason {},
RedirectingPartyID {},
RedirectionInformation,
RegistratorIdentifier,
ReportCondition,
RouteingNumber {},
RequestedInformationList {},
RequestedInformationTypeList,
RequestedUTSIList {},
ResourceID {},
ResourceStatus,
ResponseCondition,
RouteList {},
ScfID {},
SCIBillingChargingCharacteristics {},
SDSSinformation {},
ServiceInteractionIndicators {},
ServiceInteractionIndicatorsTwo,
ServiceKey,
TDPIdentifier,
TerminalType,
TimeAndTimezone {},
TimerID,
TimerValue,
TriggerData,
TriggerDataIdentifier {},
TriggerDPType,
```

```
            Triggers  {},
            TriggerStatus,
            TriggerResults {},
            USIInformation {},
            USIServiceIndicator {},
            VPNIndicator
FROM IN-CS3-SSF-SCF-datatypes ssf-scf-datatypes

    InformationToSend {}
FROM IN-CS3-scf-srf-datatypes scf-srf-datatypes

    cancelFailed,
    eTCFailed,
    improperCallerResponse,
    missingCustomerRecord,
    missingParameter,
    parameterOutOfRange,
    requestedInfoError,
    systemFailure,
    taskRefused,
    unavailableResource,
    unexpectedComponentSequence,
    unexpectedDataValue,
    unexpectedParameter,
    unknownLegID,
    unknownResource
FROM IN-CS3-errortypes errortypes
;
```

-- The following three definitions are local short-hand notation for convenience.
B1::= COMMON-BOUNDS   -- defined in EN 301 931-1
B2::= SCF-SSF-BOUNDS   -- defined in this part
B3::= SCF-SRF-BOUNDS   -- defined in EN 301 931-3

-- Operations and Arguments:

activateServiceFiltering {B1: b1, B2: b2, B3: b3} OPERATION::= {
**   ARGUMENT   ActivateServiceFilteringArg {b1, b2, b3}**
**   RETURN RESULT TRUE**
**   ERRORS {missingParameter |**
**   parameterOutOfRange |**
**   systemFailure |**
**   taskRefused |**
**   unexpectedComponentSequence |**
**   unexpectedParameter**
**   }**
**   CODE   opcode-activateServiceFiltering**
**   }**
-- Direction: SCF -> SSF, Timer: T_{asf}
-- When receiving this operation, the SSF handles calls to destination in a specified manner
-- without sending queries for every detected call. It is used for example for providing
-- televoting or mass calling services. Simple registration functionality (counters) and
-- announcement control may be located at the SSF. The operation initializes the specified
-- counters in the SSF.

ActivateServiceFilteringArg {B1: b1, B2: b2, B3: b3}::= SEQUENCE {
**   filteredCallTreatment   [0] FilteredCallTreatment {b1, b2, b3},**
**   filteringCharacteristics   [1] FilteringCharacteristics,**
**   filteringTimeOut   [2] FilteringTimeOut,**
**   filteringCriteria   [3] FilteringCriteria {b2},**
**   startTime   [4] DateAndTime   OPTIONAL,**
**   extensions   [5] Extensions {b1}   OPTIONAL,**
**   ...**
**   }**

activityTest OPERATION::= {
**   RETURN RESULT TRUE**
**   ALWAYS RESPONDS FALSE**
**   CODE   opcode-activityTest**
**   }**
-- Direction: SCF -> SSF or SSF-> SCF, Timer: T_{at}
-- This operation is used to check for the continued existence of a relationship between the SCF
-- and SSF. If the relationship is still in existence, then the SSF will respond. If no reply is
-- received, then the SCF will assume that the SSF has failed in some way and will take the
-- appropriate action.. As an option, this operation may be used in the reverse direction by the
-- SSF to check for the continued existence of a relationship with the SCF.

```
applyCharging {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT     ApplyChargingArg {b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            unexpectedComponentSequence |
            unexpectedParameter |
            unexpectedDataValue |
            parameterOutOfRange |
            systemFailure |
            taskRefused|
            unknownLegID}
    ALWAYS RESPONDS FALSE    CODE     opcode-applyCharging
    }
-- Direction: SCF ->  SSF, Timer: T_ac
-- This operation is used  for  interacting  from  the SCF with the SSF charging mechanisms.
-- The ApplyChargingReport operation provides the  feedback from the SSF to  the SCF.
-- This operation is can also be used to instruct the SSF to release the call regarding
-- some condition.

ApplyChargingArg {B1: b1, B2: b2}::= SEQUENCE {
    aChBillingChargingCharacteristics   [0] AChBillingChargingCharacteristics {b2} OPTIONAL,
    -- parameter is made optional to allow backward compatibility and to avoid a mandatory use
    -- of a CS1 defined operator specific Octet string.
    sendCalculationToSCPIndication      [1] BOOLEAN OPTIONAL,
    -- This parameter, if present, shall be set toTRUE.
-- The parameteris ignored if CS2 or CS3 is used.
    aChChargingAddress                  AChChargingAddress {b2} DEFAULT legID:sendingSideID:leg1,
    extensions                          [3] Extensions {b1}     OPTIONAL,
    callSupervision                     [52] CallSupervision {b2}   OPTIONAL,
    ...
    }

applyChargingReport {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT     ApplyChargingReportArg {b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            unexpectedComponentSequence |
            unexpectedParameter |
            unexpectedDataValue |
            parameterOutOfRange |
            systemFailure |
            taskRefused}
    ALWAYS RESPONDS FALSE
    CODE    opcode-applyChargingReport
    }

-- Direction: SSF ->  SCF, Timer: T_acr
-- This operation is used  by the SSF to  report to the SCF the occurrence of a specific
-- charging event as requested  by the SCF using the ApplyCharging operation.

ApplyChargingReportArg { B1: b1, B2: b2}::= CallResult {b1, b2}

assistRequestInstructions {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT     AssistRequestInstructionsArg {b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingCustomerRecord|
            missingParameter |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter}
    ALWAYS RESPONDS FALSE
    CODE    opcode-assistRequestInstructions
    }
-- Direction: SSF  -> SCF or SRF  -> SCF, Timer: Tari
-- This operation is used when there is an assist or a hand-off procedure and may be sent by the SSF
-- or SRF to the SCF. This operation is sent by the assisting SSF to SCF, when the initiating
-- SSF has set up a connection to the SRF or to the assisting SSF as a result of receiving
-- an EstablishTemporaryConnection or Connect operation (in the case of hand-off)
-- from the SCF.
```

```
AssistRequestInstructionsArg {B1: b1, B2: b2}::= SEQUENCE {
    correlationID       [0] CorrelationID {b2},
    iPAvailable         [1] IPAvailable {b2}         OPTIONAL,
    iPSSPCapabilities   [2] IPSSPCapabilities {b2}   OPTIONAL,
    extensions          [3] Extensions {b1}          OPTIONAL,
    ...
    }
-- OPTIONAL denotes network operator specific use. The value of the correlationID may be the
-- Called Party Number supplied by the initiating SSF.

callGap { B1: b1, B2: b2, B3: b3} OPERATION::= {
    ARGUMENT    CallGapArg {b1, b2, b3}
    RETURN RESULT    FALSE
    ALWAYS RESPONDS      FALSE
    CODE    opcode-callGap
    }
```
-- Direction: SCF -> SSF, Timer: T_{cg}
```
-- This operation is used to request the SSF  to reduce the rate at which specific
-- service requests are sent to the  SCF.
-- Use of this operation by the SCF to gap queries and updates at the SDF is outside
-- the scope of this capability set.

CallGapArg {B1: b1, B2: b2, B3: b3}::= SEQUENCE {
    gapCriteria             [0] GapCriteria {b2},
    gapIndicators           [1] GapIndicators,
    controlType             [2] ControlType OPTIONAL,
    gapTreatment            [3] GapTreatment {b1, b2, b3}   OPTIONAL,
    extensions              [4] Extensions {b1}             OPTIONAL,
    ...
    }
-- OPTIONAL denotes network operator optional. If gapTreatment is not present, the SSF will use
-- a default treatment depending on network operator implementation.


callInformationReport { B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    CallInformationReportArg { b1, b2}
    RETURN RESULT    FALSE
    ALWAYS RESPONDS FALSE
    CODE    opcode-callInformationReport
    }
```
-- Direction: SSF -> SCF, Timer: T_{cirp}
```
-- This operation is used to send specific call information for a single call to
-- the SCF as requested by the SCF
-- in a previous CallInformationRequest.

CallInformationReportArg {B1: b1, B2: b2}::= SEQUENCE {
    requestedInformationList    [0] RequestedInformationList {b2},
    extensions                  [2] Extensions {b1}     OPTIONAL,
    legID                       [3] LegID               OPTIONAL,
    ...
    }
-- OPTIONAL denotes network operator optional.

callInformationRequest {B1: b1} OPERATION::= {
    ARGUMENT    CallInformationRequestArg {b1}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            requestedInfoError |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter|
            unknownLegID}
    ALWAYS RESPONDS FALSE
    CODE    opcode-callInformationRequest
    }
```
-- Direction: SCF -> SSF, Timer: T_{cirq}
```
-- This operation is used to request the SSF to  record specific information about a
-- single call and report it to
-- the SCF (with a CallInformationReport operation).
```

```
CallInformationRequestArg {B1: b1}::= SEQUENCE {
    requestedInformationTypeList    [0] RequestedInformationTypeList,
    extensions                      [2] Extensions {b1}    OPTIONAL,
    legID                           [3] LegID   OPTIONAL,
    ...
    }
-- OPTIONAL denotes network operator optional.


cancel {B2: b2} OPERATION::= {
    ARGUMENT    CancelArg {b2}
    RETURN RESULT    FALSE
    ERRORS  {cancelFailed |
            missingParameter |
            taskRefused}
    ALWAYS RESPONDS FALSE
    CODE    opcode-cancel
    }
-- Direction: SCF ->  SSF, or SCF  -> SRF, Timer:  T_can
-- This operation cancels the correlated previous operation or all previous requests
-- This operation can also be used to cancel all outstanding requests and enable the
-- state machine (SSF)to go to idle.
-- In  this case the Cancel operation  does not specify any specific operation to be cancelled.
-- For the SCF-SRF operations that can be cancelled, refer to Part 3 of ITU-T Recommendation Q.1238


CancelArg {B2: b2}::= CHOICE {
    invokeID                        [0] InvokeID,
    allRequests                     [1] NULL,
    callSegmentToCancel             [2] SEQUENCE {
            invokeID                    [0] InvokeID,
            callSegmentID               [1] CallSegmentID {b2},
            ...
            },
    allRequestsForCallSegment       [3] CallSegmentID {b2},
    ...
    }
-- The InvokeID has the same value as that which was used for the SCF-SRF  operation,
-- i.e. is used to identify the correlated previous SCF-SRF operation to be cancelled.


cancelStatusReportRequest { B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    CancelStatusReportRequestArg {b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {cancelFailed |
            missingParameter |
            taskRefused}
    ALWAYS RESPONDS FALSE
    CODE    opcode-cancelStatusReportRequest
    }
-- Direction: SCF ->  SSF, Timer:  T_csr
-- This operation cancels the following processes: RequestFirstStatusMatchReport and
-- RequestEveryStatusChangeReport.


CancelStatusReportRequestArg {B1: b1, B2: b2}::= SEQUENCE {
    resourceID  [0] ResourceID {b2} OPTIONAL,
    extensions  [1] Extensions {b1} OPTIONAL,
    ...
    }


collectInformation {B1: b1} OPERATION::= {
    ARGUMENT    CollectInformationArg { b1} OPTIONAL TRUE
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter}
    ALWAYS RESPONDS FALSE
    CODE    opcode-collectInformation
    }
-- Direction: SCF ->  SSF, Timer: T_ci
-- This operation is used to request the SSF to perform the originating basic call
-- processing actions to prompt a calling party for destination information,
-- then collect destination information according to a specified
-- numbering plan (e.g. for virtual private networks).
```

```
CollectInformationArg {B1: b1}::= SEQUENCE {
    extensions                   [4] Extensions {b1}          OPTIONAL,
    ...
    }


connect {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    ConnectArg {b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter}
    ALWAYS RESPONDS FALSE
    CODE    opcode-connect
    }
-- Direction: SCF ->  SSF, Timer: T_con
-- This operation is used to request the SSF to perform the call processing actions to route
-- or forward a call to a specified destination. To do so, the SSF may or may not use
-- destination information from the calling party (e.g. dialed digits) and existing call
-- setup information (e.g. route index to a list of trunk groups), depending on
-- the information provided by the SCF.
-- When address information is only included in the Connect operation, call processing
-- resumes at PIC Analyse_Information in the O-BCSM.
--  When address information and routing information is included, call processing resumes at PIC
--  Select_Route.

ConnectArg {B1: b1, B2: b2}::= SEQUENCE {
    destinationRoutingAddress        [0] DestinationRoutingAddress { b2},
    alertingPattern                  [1] AlertingPattern                       OPTIONAL,
    correlationID                    [2] CorrelationID { b2}                    OPTIONAL,
    cutAndPaste                      [3] CutAndPaste                            OPTIONAL,
    iSDNAccessRelatedInformation     [5] ISDNAccessRelatedInformation {b2}      OPTIONAL,
    originalCalledPartyID            [6] OriginalCalledPartyID { b2}            OPTIONAL,
    routeList                        [7] RouteList { b2}                        OPTIONAL,
    scfID                            [8] ScfID { b2}                            OPTIONAL,
    extensions                       [10] Extensions {b1}                       OPTIONAL,
    carrier                          [11] Carrier {b2}                          OPTIONAL,
    serviceInteractionIndicators     [26] ServiceInteractionIndicators { b2}    OPTIONAL,
    callingPartyNumber               [27] CallingPartyNumber { b2}              OPTIONAL,
    callingPartysCategory            [28] CallingPartysCategory                 OPTIONAL,
    redirectingPartyID               [29] RedirectingPartyID { b2}              OPTIONAL,
    redirectionInformation           [30] RedirectionInformation                OPTIONAL,
    displayInformation               [12] DisplayInformation { b2}              OPTIONAL,
    forwardCallIndicators            [13] ForwardCallIndicators                 OPTIONAL,
    genericNumbers                   [14] GenericNumbers { b2}                  OPTIONAL,
    serviceInteractionIndicatorsTwo  [15] ServiceInteractionIndicatorsTwo       OPTIONAL,
    iNServiceCompatibilityResponse   [16] INServiceCompatibilityResponse        OPTIONAL,
    forwardGVNS                      [17] ForwardGVNS { b2}                      OPTIONAL,
    backwardGVNS                     [18] BackwardGVNS { b2}                     OPTIONAL,
    chargeNumber                     [19] ChargeNumber { b2}                     OPTIONAL,
    callSegmentID                    [20] CallSegmentID {b2}                     OPTIONAL,
    legToBeCreated                   [21] LegID                                  OPTIONAL,
    sDSSinformation                  [22] SDSSinformation { b2}                  OPTIONAL,
    calledDirectoryNumber            [23] CalledDirectoryNumber { b2}            OPTIONAL,
    bearerCapability                 [24] BearerCapability {b2}                  OPTIONAL,
    cug-Interlock                    [31] CUG-Interlock                          OPTIONAL,
    cug-OutgoingAccess               [32] NULL                                   OPTIONAL,
    suppressionOfAnnouncement        [55] SuppressionOfAnnouncement              OPTIONAL,
    oCSIApplicable                   [56] OCSIApplicable                         OPTIONAL,
    na-OliInfo                       [57] NAOliInfo                              OPTIONAL,
    ...
    }
-- na-OliInfo is included at the discretion of the gsmSCF operator.
-- OPTIONAL parameters are only provided if modifications desired to basic call
-- processing values
-- TAG 4, 9, 50 and 51 are reserved and shall not be used.
```

```
connectToResource {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    ConnectToResourceArg { b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter|
            unknownLegID}
    ALWAYS RESPONDS FALSE
    CODE    opcode-connectToResource
    }
-- Direction: SCF ->  SSF, Timer: T_ctr
-- This operation is used to connect a call from the SSP to the physical entity containing the SRF.

ConnectToResourceArg {B1: b1, B2: b2}::= SEQUENCE {
    resourceAddress CHOICE {
        ipRoutingAddress           [0] IPRoutingAddress { b2},
        legID                      [1] LegID,
        ipAddressAndLegID          [2] SEQUENCE {
            ipRoutingAddress             [0] IPRoutingAddress {b2},
            legID                        [1] LegID,
            ...
            },
        none                       [3] NULL,
        callSegmentID              [5] CallSegmentID { b2},
        ipAddressAndCallSegment    [6] SEQUENCE {
            ipRoutingAddress             [0] IPRoutingAddress {b2},
            callSegmentID                [1] CallSegmentID { b2},
            ...
            }
        },
    extensions                         [4]  Extensions {b1}                          OPTIONAL,
    serviceInteractionIndicators       [30] ServiceInteractionIndicators { b2}       OPTIONAL,
    serviceInteractionIndicatorsTwo [7] ServiceInteractionIndicatorsTwo             OPTIONAL,
    uSIServiceIndicator                [35] USIServiceIndicator { b2}                OPTIONAL,
    uSIInformation                     [36] USIInformation { b2}                     OPTIONAL,
    ...
    }

continue OPERATION::= {
    RETURN RESULT    FALSE
    ALWAYS RESPONDS FALSE
    CODE    opcode-continue
    }
-- Direction: SCF ->  SSF, Timer: T_cue
-- This operation is used to request the SSF to proceed with call processing at the DP at which it
-- previously suspended call processing to await SCF instructions (i.e. proceed to the next point
-- in call in the BCSM). The SSF continues call processing without substituting new data from SCF.
-- This operation is not valid for a single call segment CSA with more than 2 legs or a
-- multi call segment CSA.

continueWithArgument {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    ContinueWithArgumentArg { b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            unexpectedComponentSequence |
            unexpectedParameter |
            unexpectedDataValue|
            unknownLegID
            }

    ALWAYS RESPONDS FALSE
    CODE    opcode-continueWithArgument
    }
-- Direction: SCF ->  SSF, Timer: T_cwa
-- This operation is used to request the SSF to proceed with call processing at the DP at
-- which it previously suspended call processing to await SCF instructions.
-- It is also used to provide additional service related information to a
-- User (Called Party or Calling Party) whilst the call processing proceeds.
```

```
ContinueWithArgumentArg {B1: b1, B2: b2}::= SEQUENCE {
    legorCSID   CHOICE{
        legID                           [0] LegID,
        csID                            [9] CallSegmentID {b2}}   DEFAULT legID:sendingSideID:leg1,
    alertingPattern                     [1] AlertingPattern                      OPTIONAL,
    genericName                         [2] GenericName { b2}                    OPTIONAL,
    iNServiceCompatibilityResponse      [3] INServiceCompatibilityResponse       OPTIONAL,
    forwardGVNS                         [4] ForwardGVNS { b2}                     OPTIONAL,
    backwardGVNS                        [5] BackwardGVNS { b2}                    OPTIONAL,
    extensions                          [6]  Extensions {b1}                      OPTIONAL,
    serviceInteractionIndicatorsTwo     [7] ServiceInteractionIndicatorsTwo       OPTIONAL,
    sDSSinformation                     [8] SDSSinformation { b2}                  OPTIONAL,
    iSDNAccessRelatedInformation        [19] ISDNAccessRelatedInformation{b2}      OPTIONAL,
    originalCalledPartyID               [10] OriginalCalledPartyID {b2}            OPTIONAL,
    callingPartyNumber                  [11] CallingPartyNumber {b2}              OPTIONAL,
    callingPartysCategory               [12] CallingPartysCategory                OPTIONAL,
    redirectingPartyID                  [13] RedirectingPartyID {b2}              OPTIONAL,
    redirectionInformation              [14] RedirectionInformation               OPTIONAL,
    forwardCallIndicators               [15] ForwardCallIndicators                OPTIONAL,
    genericNumbers                      [16] GenericNumbers {b2}                  OPTIONAL,
    cug-Interlock                       [17] CUG-Interlock                        OPTIONAL,
    cug-OutgoingAccess                  [18] NULL                                 OPTIONAL,
    chargeNumber                        [50] ChargeNumber { b2}                   OPTIONAL,
    carrier                             [52] Carrier { b2}                        OPTIONAL,
    suppressionOfAnnouncement           [55] SuppressionOfAnnouncement            OPTIONAL,
    na-OliInfo                          [56] NAOliInfo                            OPTIONAL,
    ...
    }
-- OPTIONAL parameters are only provided if modifications desired to basic call processing values

-- Tag value 51 is reserved.

createCallSegmentAssociation {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    CreateCallSegmentAssociationArg { b1} OPTIONAL TRUE
    RESULT  CreateCallSegmentAssociationResultArg { b1,  b2}
    ERRORS  {missingParameter |
            systemFailure|
            taskRefused|
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter
            }
    CODE    opcode-createCallSegmentAssociation
    }
-- Direction SCF -> SSF, Timer T_csa
-- This operation is used to create a new CSA. The new CSA will not contain any Call Segments
-- after creation.
-- The SSF is responsible for specifying a new CSA identifier for the created CSA which is
-- unique within the SSF.

CreateCallSegmentAssociationArg {B1: b1}::= SEQUENCE {
    extensions                  [0] Extensions {b1} OPTIONAL,
    ...
    }

CreateCallSegmentAssociationResultArg {B1:b1, B2: b2}::= SEQUENCE {
    newCallSegmentAssociation   [0] CSAID { b2},
    extensions                  [1] Extensions {b1} OPTIONAL,
    ...
    }

createOrRemoveTriggerData {B1: b1, B2: b2, B3: b3} OPERATION::= {
    ARGUMENT    CreateOrRemoveTriggerDataArg {b1, b2, b3}
    RESULT  CreateOrRemoveTriggerDataResultArg {b1, b2}
    ERRORS  {missingParameter |
            missingCustomerRecord |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter
            }
    CODE        opcode-createOrRemoveTriggerData
    }
```

```
-- Direction: SCF -> SSF, Class 1, Timer: Tcrt
-- This trigger management operation is used by the SCF outside the context of a call
-- to create a new trigger detection point in the CCF/SSF by downloading trigger data
-- (e.g. triggering criteria, ServiceKey, SCF address,....)
-- or to remove an existing trigger.


CreateOrRemoveTriggerDataArg {B1: b1, B2: b2, B3: b3} ::= SEQUENCE {
    createOrRemove        [0] CreateOrRemoveIndicator DEFAULT create,
    dPName                [1] EventTypeBCSM                        OPTIONAL,
    triggerDPType         [2] TriggerDPType   DEFAULT tdp-r,
    serviceKey            [3]  ServiceKey                          OPTIONAL,
    profile               [4] ProfileIdentifier{b2}               OPTIONAL,
    triggerData           [5] TriggerData                         OPTIONAL,
    defaultFaultHandling  [6] DefaultFaultHandling {b1, b2, b3}    OPTIONAL,
    tDPIdentifier         [7] TDPIdentifier {b2}                   OPTIONAL,
    ...,
    ...,
    extensions            [30] Extensions {b1}                     OPTIONAL
    }

CreateOrRemoveTriggerDataResultArg {B1: b1, B2: b2} ::= SEQUENCE {
    triggerStatus         [0] TriggerStatus,
    tDPIdentifier         [1] TDPIdentifier {b2},
    registratorIdentifier [2] RegistratorIdentifier                OPTIONAL,
    ...,
    ...,
    extensions            [30]  Extensions {b1}                    OPTIONAL
    }

disconnectForwardConnection OPERATION::= {
    RETURN RESULT    FALSE
    ERRORS  {systemFailure |
            taskRefused |
            unexpectedComponentSequence }
    ALWAYS RESPONDS FALSE
    CODE    opcode-disconnectForwardConnection
    }
-- Direction: SCF -> SSF, Timer: T_dfc
-- This operation is used to disconnect a forward temporary connection or a
-- connection to a resource.

disconnectForwardConnectionWithArgument {B1: b1,B2: b2} OPERATION::= {
    ARGUMENT    DisconnectForwardConnectionWithArgumentArg {b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter |
            unknownLegID}
    ALWAYS RESPONDS FALSE
    CODE    opcode-dFCWithArgument
    }

-- Direction: SCF -> SSF, Timer: T_dfcwa
-- This operation is used to disconnect a forward temporary connection or a
-- connection to a resource.

DisconnectForwardConnectionWithArgumentArg {B1: b1,B2: b2}::= SEQUENCE {
    partyToDisconnect   CHOICE {
            legID               [0] LegID,
            callSegmentID       [1] CallSegmentID { b2}
            },
    extensions              [2] Extensions {b1}                    OPTIONAL,
    uSISServiceIndicator    [3] USISServiceIndicator { b2}         OPTIONAL,
    uSIInformation          [4] USIInformation { b2}               OPTIONAL,
    ...
    }
```

```
disconnectLeg {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    DisconnectLegArg { b1, b2}
    RETURN RESULT TRUE
    ERRORS  {missingParameter|
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter|
            unknownLegID}
    CODE    opcode-disconnectLeg
    }
-- Direction: SCF -> SSF. Timer: T dl
-- This operation  is issued by the SCF  to release a specific leg associated with the call
-- and retain any other legs not specified in the DisconnectLeg. Any leg may be disconnected,
-- including the controlling leg, without completely releasing all legs.

DisconnectLegArg {B1: b1, B2: b2}::= SEQUENCE {
    legToBeReleased     [0] LegID,
    releaseCause        [1] Cause { b2}              OPTIONAL,
    extensions          [2] Extensions {b1}         OPTIONAL,
    ...
    }

entityReleased {B2: b2}  OPERATION::= {
    ARGUMENT    EntityReleasedArg { b2}
    RETURN RESULT    FALSE
    ALWAYS RESPONDS      FALSE
    CODE    opcode-entityReleased
    }

-- Direction SSF -> SCF, Timer: T_er
-- This operation is used by SSF to inform the SCF of an error/exception

EntityReleasedArg {B2: b2} ::= CHOICE {
    cSFailure   [0] SEQUENCE{
            callSegmentID   [0] CallSegmentID { b2},
            reason          [1] Reason { b2}         OPTIONAL,
            cause           [2] Cause { b2}          OPTIONAL,
            ...
            },
    bCSMFailure [1] SEQUENCE{
            legID           [0] LegID,
            reason          [1] Reason { b2}         OPTIONAL,
            cause           [2] Cause { b2}          OPTIONAL,
            ...
            }
    }

establishTemporaryConnection {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    EstablishTemporaryConnectionArg { b1, b2}
    RETURN RESULT    FALSE
    ERRORS {eTCFailed |
            missingParameter |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter|
            unknownLegID}
    ALWAYS RESPONDS FALSE
    CODE    opcode-establishTemporaryConnection
    }
-- Direction: SCF ->  SSF, Timer: T_etc
-- This operation is used to create a connection to a resource for a limited period of time
-- (e.g. to play an announcement, to collect user information); it implies the use of the assist
-- procedure.
```

```
EstablishTemporaryConnectionArg {B1: b1, B2: b2}::= SEQUENCE {
    assistingSSPIPRoutingAddress      [0] AssistingSSPIPRoutingAddress { b2},
    correlationID                     [1] CorrelationID { b2}                OPTIONAL,
    partyToConnect  CHOICE {
                    legID             [2] LegID,
                    callSegmentID     [7] CallSegmentID { b2}
                    }                                                        OPTIONAL,
    scfID                             [3] ScfID { b2}                        OPTIONAL,
    extensions                        [4] Extensions {b1}                    OPTIONAL,
    carrier                           [5] Carrier {b2}                       OPTIONAL,
    serviceInteractionIndicators      [30] ServiceInteractionIndicators { b2} OPTIONAL,
    serviceInteractionIndicatorsTwo [6] ServiceInteractionIndicatorsTwo      OPTIONAL,
    na-OliInfo                        [50] NAOliInfo                         OPTIONAL,
    chargeNumber                      [51] ChargeNumber {b2}                 OPTIONAL,
    ...
    }
-- OPTIONAL parameters are only provided if modifications desired to basic call processing values

eventNotificationCharging {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT     EventNotificationChargingArg { b1, b2}
    RETURN RESULT    FALSE
    ALWAYS RESPONDS FALSE
    CODE     opcode-eventNotificationCharging
    }
```
-- Direction: SSF -> SCF, Timer: T_{enc}
-- This operation is used by the SSF to report to the SCF the occurrence of a specific
-- charging event
-- type as   previously requested by the SCF in a
-- RequestNotificationChargingEvent operation.

```
EventNotificationChargingArg {B1: b1, B2: b2}::= SEQUENCE {
    eventTypeCharging                 [0] EventTypeCharging { b2} OPTIONAL,
    eventSpecificInformationCharging  [1] EventSpecificInformationCharging { b2}  OPTIONAL,
    legID                             [2] LegID                                OPTIONAL,
    extensions                        [3] Extensions {b1}                       OPTIONAL,
    monitorMode                       [30] MonitorMode    DEFAULT notifyAndContinue,
    eventTypeTariff                   [50] EventTypeTariff                      OPTIONAL,
    eventSpecificInformationTariff    [51] ChargingMessageType                  OPTIONAL,
    ...
    }
```
-- OPTIONAL denotes network operator specific use.

```
eventReportBCSM {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT     EventReportBCSMArg { b1, b2}
    RETURN RESULT    FALSE
    ALWAYS RESPONDS FALSE
    CODE     opcode-eventReportBCSM
    }
```
-- Direction: SSF -> SCF, Timer: T_{erb}
-- This operation is used to notify the SCF of a call-related event (e.g. BCSM events such as
-- busy or no answer) previously requested by the SCF in a RequestReportBCSMEvent operation.

```
EventReportBCSMArg {B1: b1, B2: b2}::= SEQUENCE {
    eventTypeBCSM                     [0] EventTypeBCSM,
        eventSpecificInformationBCSM[2] EventSpecificInformationBCSM { b2}  OPTIONAL,
    legID                             [3] LegID                             OPTIONAL,
    miscCallInfo                      [4] MiscCallInfo    DEFAULT {messageType request},
    extensions                    [5] Extensions {b1}                  OPTIONAL,
    ...
    }

furnishChargingInformation {B1:b1, B2: b2} OPERATION::= {
    ARGUMENT     FurnishChargingInformationArg { b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter}
    ALWAYS RESPONDS FALSE
    CODE     opcode-furnishChargingInformation
    }
```
-- Direction: SCF -> SSF, Timer: T_{fci}
-- This operation is used to request the SSF to generate, register a call record or to
-- include some information in the default call record. The registered call record is intended
-- for off line charging of the call.

```
FurnishChargingInformationArg {B1: b1, B2: b2}::= FCIBillingChargingCharacteristics {b1, b2}


initialDP {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    InitialDPArg { b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingCustomerRecord |
           missingParameter |
           parameterOutOfRange |
           systemFailure |
           taskRefused |
           unexpectedComponentSequence |
           unexpectedDataValue |
           unexpectedParameter
           }
    ALWAYS RESPONDS FALSE
    CODE    opcode-initialDP
    }
-- Direction: SSF ->  SCF, Timer: T_idp
-- This operation is used after a TDP to indicate request for service.


InitialDPArg {B1: b1, B2: b2}::= SEQUENCE {
    serviceKey                      [0] ServiceKey,
    calledPartyNumber               [2] CalledPartyNumber { b2}              OPTIONAL,
    callingPartyNumber              [3] CallingPartyNumber { b2}             OPTIONAL,
    callingPartyBusinessGroupID     [4] CallingPartyBusinessGroupID          OPTIONAL,
    callingPartysCategory           [5] CallingPartysCategory                OPTIONAL,
    cGEncountered                   [7] CGEncountered                        OPTIONAL,
    iPSSPCapabilities               [8] IPSSPCapabilities { b2}              OPTIONAL,
    iPAvailable                     [9] IPAvailable { b2}                    OPTIONAL,
    locationNumber                  [10] LocationNumber { b2}                OPTIONAL,
    originalCalledPartyID            [12] OriginalCalledPartyID {b2}          OPTIONAL,
    terminalType                    [14] TerminalType                        OPTIONAL,
    extensions                      [15] Extensions {b1}                     OPTIONAL,
    highLayerCompatibility          [23] HighLayerCompatibility              OPTIONAL,
    serviceInteractionIndicators    [24] ServiceInteractionIndicators { b2} OPTIONAL,
    additionalCallingPartyNumber    [25] AdditionalCallingPartyNumber { b2} OPTIONAL,
    forwardCallIndicators           [26] ForwardCallIndicators               OPTIONAL,
    bearerCapability                [27] BearerCapability { b2}              OPTIONAL,
    eventTypeBCSM                   [28] EventTypeBCSM                        OPTIONAL,
    redirectingPartyID              [29] RedirectingPartyID { b2}            OPTIONAL,
    redirectionInformation          [30] RedirectionInformation              OPTIONAL,
    cause                           [17] Cause { b2}                         OPTIONAL,
    iSDNAccessRelatedInformation    [21] ISDNAccessRelatedInformation { b2} OPTIONAL,
    iNServiceCompatibilityIndication [22] INServiceCompatibilityIndication { b2}
                                                                            OPTIONAL,
    genericNumbers                  [31] GenericNumbers { b2}                OPTIONAL,
    serviceInteractionIndicatorsTwo [32] ServiceInteractionIndicatorsTwo    OPTIONAL,
    forwardGVNS                     [33] ForwardGVNS { b2}                   OPTIONAL,
    createdCallSegmentAssociation   [34] CSAID { b2}                         OPTIONAL,
    uSIServiceIndicator             [35] USIServiceIndicator { b2}           OPTIONAL,
    uSIInformation                  [36] USIInformation { b2}                OPTIONAL,
    carrier                         [37] Carrier {b2}                        OPTIONAL,
    cCSS                            [38] CCSS                                OPTIONAL,
    vPNIndicator                    [39] VPNIndicator                        OPTIONAL,
    cNInfo                          [40] CNInfo { b2}                        OPTIONAL,
    callReference                   [41] CallReference{ b2}                  OPTIONAL,
    routeingNumber                  [42] RouteingNumber { b2}                OPTIONAL,
    callingGeodeticLocation         [43] CallingGeodeticLocation { b2}       OPTIONAL,
    globalCallReference             [44] GlobalCallReference {b2}            OPTIONAL,
    cug-Index                       [45] CUG-Index                           OPTIONAL,
    cug-Interlock                   [46] CUG-Interlock                       OPTIONAL,
    cug-OutgoingAccess              [47] NULL                                OPTIONAL,
    iMSI                            [50] IMSI                                OPTIONAL,
    subscriberState                 [51] SubscriberState                     OPTIONAL,
    locationInformation             [52] LocationInformation                 OPTIONAL,
    ext-basicServiceCode            [53] Ext-BasicServiceCode                OPTIONAL,
    callReferenceNumber             [54] CallReferenceNumber                 OPTIONAL,
    mscAddress                      [55] ISDN-AddressString                  OPTIONAL,
    calledPartyBCDNumber            [56] CalledPartyBCDNumber    {b2}        OPTIONAL,
    timeAndTimezone                 [57] TimeAndTimezone {b2}                OPTIONAL,
    gsm-ForwardingPending           [58] NULL                                OPTIONAL,
    initialDPArgExtension           [59] InitialDPArgExtension               OPTIONAL,
    ...
    }
```

```
-- for mscAddress for encoding see TS 129 002
-- OPTIONAL for iPSSPCapabilities, iPAvailable, cGEncountered, and miscCallInfo denotes network
-- operator specific use.
-- OPTIONAL for terminalType indicates that this parameter applies only at originating
-- or terminating
-- local exchanges if the SSF has this information.
-- Tag values 1, 6, 11, 13, 16, 18, 19, 20 are reserved and shall not be used


initiateCallAttempt {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    InitiateCallAttemptArg { b1, b2}
    RETURN RESULT   FALSE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter
            }
    ALWAYS RESPONDS FALSE

    CODE    opcode-initiateCallAttempt
    }
-- Direction: SCF ->  SSF, Timer: T_ica
-- This operation is used to request the SSF to create a new call to one call  party using address
-- information provided by the SCF.
InitiateCallAttemptArg {B1: b1, B2: b2}::= SEQUENCE {
    destinationRoutingAddress     [0] DestinationRoutingAddress { b2},
    alertingPattern               [1] AlertingPattern                      OPTIONAL,
    iSDNAccessRelatedInformation  [2] ISDNAccessRelatedInformation {b2}     OPTIONAL,
    extensions                    [4]  Extensions {b1}                      OPTIONAL,
    serviceInteractionIndicators  [29] ServiceInteractionIndicators { b2}   OPTIONAL,
    callingPartyNumber            [30] CallingPartyNumber { b2}             OPTIONAL,
    legToBeCreated                [5] LegID   DEFAULT      sendingSideID:leg1,
    newCallSegment                [6] CallSegmentID { b2} DEFAULT initialCallSegment,
    iNServiceCompatibilityResponse [7] INServiceCompatibilityResponse      OPTIONAL,
    serviceInteractionIndicatorsTwo [8] ServiceInteractionIndicatorsTwo    OPTIONAL,
    carrier                       [9] Carrier {b2}                          OPTIONAL,
    correlationID                 [10] CorrelationID { b2}                  OPTIONAL,
    scfID                         [11] ScfID { b2}                          OPTIONAL,
    callReference                 [12] CallReference { b2}                  OPTIONAL,
    calledDirectoryNumber         [13] CalledDirectoryNumber { b2}          OPTIONAL,
    originalCalledPartyID         [14] OriginalCalledPartyID {b2}           OPTIONAL,
    callingPartysCategory         [15] CallingPartysCategory               OPTIONAL,
    redirectingPartyID            [16] RedirectingPartyID {b2}              OPTIONAL,
    redirectionInformation        [17] RedirectionInformation              OPTIONAL,
    displayInformation            [18] DisplayInformation {b2}              OPTIONAL,
    forwardCallIndicators         [19] ForwardCallIndicators               OPTIONAL,
    genericNumbers                [20] GenericNumbers {b2}                  OPTIONAL,
    forwardGVNS                   [21] ForwardGVNS {b2}                     OPTIONAL,
    bearerCapability              [60] BearerCapability { b2}               OPTIONAL,
    globalCallReference           [23] GlobalCallReference { b2}            OPTIONAL,
    cug-Interlock                 [24] CUG-Interlock                        OPTIONAL,
    cug-OutgoingAccess            [25] NULL                                 OPTIONAL,
    ...
    }
-- Tag values 3, 50, 51, 52 are reserved and shall not be used
-- OPTIONAL parameters are only provided if modifications desired to basic call processing values

manageTriggerData {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    ManageTriggerDataArg { b1, b2}
    RESULT  ManageTriggerDataResultArg { b1, b2}
    ERRORS  {missingParameter |
            missingCustomerRecord |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter
            }
    CODE    opcode-manageTriggerData
    }
```

```
-- Direction: SCF  -> SSF,  Class 1, Timer: T_mtd
-- This trigger management operation is used outside the context of a call to activate,
-- deactivate or retrieve
-- the status of one or several trigger detection point linked to a subscriber profile known at
-- the switch, e.g. related to an access line ( i.e. an individual trigger).


ManageTriggerDataArg {B1: b1, B2: b2} ::= SEQUENCE {
    actionIndicator        [0] ActionIndicator,
    triggerDataIdentifier   CHOICE {
            profileAndDP            [1] TriggerDataIdentifier {b1, b2},
            -- one trigger
            profile                 [5] ProfileIdentifier {b2}
        },
    registratorIdentifier  [2] RegistratorIdentifier             OPTIONAL,
    extensions             [3] Extensions {b1}                   OPTIONAL,
    tDPIdentifier          [4] TDPIdentifier {b2}                OPTIONAL,
    ...
    }

ManageTriggerDataResultArg {B1: b1, B2: b2} ::= CHOICE
    {
    oneTriggerResult            SEQUENCE {
                actionPerformed     [0]  ActionPerformed,
                extensions          [1] Extensions {b1}          OPTIONAL,
    ... },
    severalTriggerResult    [1] SEQUENCE {
                results             [0] TriggerResults {b2},
                extensions          [1] Extensions {b1}          OPTIONAL,
    ... }
    }

mergeCallSegments {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    MergeCallSegmentsArg { b1, b2}
    RETURN RESULT TRUE
    ERRORS  {missingParameter |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter
            }
    CODE    opcode-mergeCallSegments
    }
-- Direction: SCF -> SSF. Timer: T _mc
-- This operation  is issued by the SCF  to merge two associated CSs, into one CS.

MergeCallSegmentsArg {B1: b1, B2: b2}::= SEQUENCE {
    sourceCallSegment          [0] CallSegmentID {b2},
    targetCallSegment          [1] CallSegmentID {b2} DEFAULT initialCallSegment,
    extensions                 [2] Extensions {b1}              OPTIONAL,
    ...
    }

moveCallSegments {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    MoveCallSegmentsArg { b1, b2}
    RETURN RESULT TRUE
    ERRORS  {missingParameter |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter|
            unknownLegID
            }
    CODE    opcode-moveCallSegments
    }
-- Direction: SCF -> SSF, Timer T_mcs
-- This operation moves a CS from the source CSA to the target CSA

MoveCallSegmentsArg {B1: b1, B2: b2}::= SEQUENCE {
    targetCallSegmentAssociation   [0] CSAID { b2},
    callSegments                   [1] SEQUENCE SIZE (1..b2.&numOfCSs)
 OF SEQUENCE {
        sourceCallSegment              [0] CallSegmentID { b2}     DEFAULT initialCallSegment,
        newCallSegment                 [1] CallSegmentID { b2},
        ...
        },
```

```
        legs                            [2] SEQUENCE SIZE (1..b2.&numOfLegs)
OF SEQUENCE {
            sourceLeg                   [0] LegID,
            newLeg                      [1] LegID,
            ...
            },
        extensions                      [3]  Extensions {b1}           OPTIONAL,
        ...
        }

moveLeg {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    MoveLegArg { b1, b2}
    RETURN RESULT TRUE
    ERRORS  {missingParameter |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter|
            unknownLegID
            }
    CODE    opcode-moveLeg
    }
-- Direction: SCF  ->SSF, Timer: T ml
-- This operation is issued by the SCF to move a leg from one CS to another with which
-- it is associated.

MoveLegArg {B1: b1, B2: b2}::=SEQUENCE {
    legIDToMove             [0] LegID,
    targetCallSegment       [1] CallSegmentID { b2} DEFAULT 1,
    extensions              [2] Extensions {b1}     OPTIONAL,
    ...
    }


releaseCall { B2: b2}  OPERATION::= {
    ARGUMENT    ReleaseCallArg { b2}
    RETURN RESULT   FALSE
    ALWAYS RESPONDS FALSE
    CODE    opcode-releaseCall
    }
-- Direction: SCF ->  SSF, Timer: T rc
-- This operation is used by the SCF to tear down an existing call segment at any phase of
-- the call for all parties involved in the call segment or to tear down all existing
-- call segments within a Call Segment Association.

ReleaseCallArg {B2: b2}::= CHOICE {
    initialCallSegment              Cause { b2},
    callSegmentToRelease    [1] SEQUENCE {
        callSegment             [0] INTEGER (1..b2.&numOfCSs),
        releaseCause            [1] Cause { b2}             OPTIONAL,
        forcedRelease           [2] BOOLEAN  DEFAULT  FALSE,
        ...
        },
    allCallSegments         [2] SEQUENCE {
        releaseCause            [0] Cause { b2}             OPTIONAL,
        timeToRelease           [1] TimerValue              OPTIONAL,
        forcedRelease           [2] BOOLEAN  DEFAULT  FALSE,
        ...
        },
    ...
    }
-- A default cause value of decimal 31 (normal unspecified) should be coded appropriately.
-- If timeToRelease parameter is omitted, the default  shall be no timed disconnect  requested
-- If forcedRelease parameter is omitted (default value  "FALSE") the default shall be no
-- forced release requested.

reportUTSI {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    ReportUTSIArg  { b1, b2}
    RETURN RESULT   FALSE
    ALWAYS RESPONDS FALSE
    CODE    opcode-reportUTSI
    }
-- Direction: SSF -> SCF. Timer: T ru
-- This operation is issued by the SSF in the context of the USI  feature.
-- It is used to report the receipt of a User to Service Information (USI)  to the SCF.
```

```
ReportUTSIArg {B1: b1, B2: b2}::= SEQUENCE {
    uSIServiceIndicator [0] USIServiceIndicator { b2},
    legID               [1] LegID   DEFAULT receivingSideID:leg1,
    uSIInformation      [2] USIInformation { b2},
    extensions          [3] Extensions {b1}                OPTIONAL,
    ...
    }

requestCurrentStatusReport {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    RequestCurrentStatusReportArg { b2}
    RESULT  RequestCurrentStatusReportResultArg { b1, b2}
    ERRORS  {missingParameter |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter |
            unknownResource
            }
    CODE    opcode-requestCurrentStatusReport
    }
-- Direction: SCF ->  SSF, Timer: T_{rcs}
-- This operation is used to request the SSF to report immediately the busy/idle status
-- of a physical termination resource.

RequestCurrentStatusReportArg {B2: b2}::= ResourceID { b2}

RequestCurrentStatusReportResultArg {B1: b1, B2: b2}::= SEQUENCE {
    resourceStatus  [0] ResourceStatus,
    resourceID      [1] ResourceID { b2}    OPTIONAL,
    extensions      [2] Extensions {b1}     OPTIONAL,
    ...
    }

requestEveryStatusChangeReport {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    RequestEveryStatusChangeReportArg { b1, b2}
    RETURN RESULT TRUE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter |
            unknownResource
            }
    CODE    opcode-requestEveryStatusChangeReport
    }
-- Direction: SCF ->  SSF, Timer: T_{res}
-- This operation is used to request the SSF to report every change of busy/idle status
-- of a physical termination resource.

RequestEveryStatusChangeReportArg {B1: b1, B2: b2}::= SEQUENCE {
    resourceID      [0] ResourceID { b2},
    correlationID   [1] CorrelationID { b2}         OPTIONAL,
    monitorDuration [2] Duration                    OPTIONAL,
    extensions      [3] Extensions {b1}             OPTIONAL,
    ...
    }
-- For correlationID OPTIONAL denotes network operator optional.
-- monitorDuration is required if outside the context of a call.
-- It is not expected if we are in the context of a call, because in that case
-- the end of the call implicitly means the end of the monitoring.

requestFirstStatusMatchReport {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    RequestFirstStatusMatchReportArg { b1, b2}
    RETURN RESULT    TRUE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter |
            unknownResource
            }
    CODE    opcode-requestFirstStatusMatchReport
```

```
    }
-- Direction: SCF ->  SSF, Timer: T_rfs
-- This operation is used to request the SSF to report the first change busy/idle to
-- the specified status of a physical termination resource.

RequestFirstStatusMatchReportArg {B1: b1, B2: b2}::= SEQUENCE {
    resourceID          [0] ResourceID { b2}            OPTIONAL,
    resourceStatus      [1] ResourceStatus  OPTIONAL,
    correlationID       [2] CorrelationID { b2}         OPTIONAL,
    monitorDuration     [3] Duration                    OPTIONAL,
    extensions          [4] Extensions {b1}             OPTIONAL,
    bearerCapability    [5] BearerCapability { b2}      OPTIONAL,
    ...
    }
-- For correlationID OPTIONAL denotes network operator optional.
-- monitorDuration is required if outside the context of a call. It is not expected if we are in
-- the context of a call, because in that case the end of the call implicitly means the end of
-- the monitoring.

requestNotificationChargingEvent {B2: b2}  OPERATION::= {
    ARGUMENT    RequestNotificationChargingEventArg { b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter |
            unknownLegID
            }
    ALWAYS RESPONDS FALSE
    CODE    opcode-requestNotificationChargingEvent
    }
-- Direction: SCF ->  SSF, Timer: T_rnc
-- This operation  is used by the SCF to instruct the SSF on how to manage the charging events
-- which are received  from other FE's and not under control of the service logic instance.

RequestNotificationChargingEventArg {B2: b2}::= SEQUENCE SIZE(1..b2.&numOfChargingEvents) OF
        ChargingEvent {b2}

requestReportBCSMEvent {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    RequestReportBCSMEventArg { b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter |
            unknownLegID

            }
    ALWAYS RESPONDS FALSE

    CODE    opcode-requestReportBCSMEvent
    }
-- Direction: SCF ->  SSF, Timer: T_rrb


RequestReportBCSMEventArg {B1: b1, B2: b2}::= SEQUENCE {
    bcsmEvents              [0] SEQUENCE SIZE(1..b2.&numOfBCSMEvents)  OF
                                BCSMEvent {b2},
        extensions          [2]  Extensions {b1}   OPTIONAL,
    ...
    }
-- Indicates the BCSM related events for notification.
```

```
requestReportUTSI  {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    RequestReportUTSIArg { b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {
            missingParameter |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter|
            unknownLegID
            }
    ALWAYS RESPONDS FALSE
    CODE    opcode-requestReportUTSI
    }
```

-- Direction: SCF -> SSF. Timer: T_{rru}
-- This operation is issued by the SCF in the context of the USI feature to request the
-- SSF to monitor for
-- a User to Service Information (UTSI) information element, which are received from a user.

```
RequestReportUTSIArg {B1: b1, B2: b2}::= SEQUENCE {
    requestedUTSIList   [0] RequestedUTSIList { b2},
    extensions          [1] Extensions {b1}            OPTIONAL,
    legID               [2] LegID   DEFAULT sendingSideID:leg1  ,
    ...
    }
```

```
resetTimer {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    ResetTimerArg { b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter
            }
    ALWAYS RESPONDS FALSE
    CODE    opcode-resetTimer
    }
```
-- Direction: SCF -> SSF, Timer: T_{rt}
-- This operation is used to request the SSF to refresh an application timer in the SSF.

```
ResetTimerArg {B1: b1, B2: b2}::= SEQUENCE {
    timerID         [0] TimerID          DEFAULT tssf,
    timervalue      [1] TimerValue,
    extensions      [2] Extensions {b1}            OPTIONAL,
    callSegmentID   [3] CallSegmentID { b2}        OPTIONAL,
    ...
    }
```

```
selectFacility {B1: b1 }  OPERATION::= {
    ARGUMENT    SelectFacilityArg { b1 }
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedComponentSequence |
            unexpectedDataValue |
            unexpectedParameter
            }
    ALWAYS RESPONDS FALSE
    CODE    opcode-selectFacility
    }
```
-- Direction: SCF -> SSF, Timer: T_{sf}
-- This operation is used to request the SSF to perform the terminating basic call processing
-- actions to select the terminating line if it is idle,. If no idle line, the SSF
-- determines that the terminating facility is busy.

```
SelectFacilityArg {B1: b1}::= SEQUENCE {
    alertingPattern                 [0] AlertingPattern                     OPTIONAL,
    extensions                      [6] Extensions {b1}                     OPTIONAL,
    serviceInteractionIndicatorsTwo [12] ServiceInteractionIndicatorsTwo    OPTIONAL,
    legID                           [16] LegID                              OPTIONAL,
```

ETSI

```
    ...
    }
-- OPTIONAL parameters are only provided if modifications desired to basic call processing values.

sendChargingInformation {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    SendChargingInformationArg { b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            unexpectedComponentSequence |
            unexpectedParameter |
            parameterOutOfRange |
            systemFailure |
            taskRefused |
            unexpectedDataValue|
            unknownLegID


            }
    ALWAYS RESPONDS FALSE
    CODE    opcode-sendChargingInformation
    }
-- Direction: SCF ->  SSF, Timer: T_{sci}
-- This operation is used to  instruct the SSF on the charging information to send by the SSF.
-- The charging information can either be sent back by means of signalling or internal
-- if  the SSF is located in the local exchange. In the local exchange
-- this information may be used to update the charge meter or to create a standard call record.

SendChargingInformationArg {B1: b1, B2: b2}::= SEQUENCE {
    sCIBillingChargingCharacteristics   [0] SCIBillingChargingCharacteristics {b2} OPTIONAL,
    partyToCharge                       [1] LegID,
    extensions                          [2] Extensions  {b1}
    OPTIONAL,
    tariffMessage                       [50] ChargingMessageType        OPTIONAL,
    chargingControl                     [51] ChargingControlType DEFAULT
                    {tariffInfoFromSuccExchangeSCI { }},
    ...
    }

sendSTUI  {B1: b1, B2: b2} OPERATION::= {
    ARGUMENT    SendSTUIArg { b1, b2}
    RETURN RESULT    FALSE
    ERRORS  {missingParameter |
            parameterOutOfRange |
            unexpectedComponentSequence |
            unexpectedParameter |
            unexpectedDataValue |
            systemFailure |
            taskRefused |
            unknownLegID
            }
    ALWAYS RESPONDS FALSE
    CODE    opcode-sendSTUI
    }
-- Direction: SCF -> SSF. Timer: T_{ss}
-- This operation is issued by the SCF in the context of the USI feature.
-- It is used to request the SSF to send a Service to User Information (USI  information)
-- data element to the indicated user.

SendSTUIArg {B1: b1, B2: b2}::= SEQUENCE {
    uSIServiceIndicator [0] USIServiceIndicator { b2},
    legID               [1] LegID   DEFAULT sendingSideID:leg1,
    uSIInformation      [2] USIInformation { b2},
    extensions          [3] Extensions {b1}              OPTIONAL,
    ...
    }

serviceFilteringResponse {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    ServiceFilteringResponseArg { b1, b2}
    RETURN RESULT    FALSE
    ALWAYS RESPONDS FALSE
    CODE    opcode-serviceFilteringResponse
    }
-- Direction: SSF ->  SCF, Timer: T_{sfr}
-- This operation is used to send back to the SCF the values of counters specified in a previous
-- ActivateServiceFiltering operation
```

```
ServiceFilteringResponseArg {B1: b1, B2: b2}::= SEQUENCE {
    countersValue       [0] CountersValue,
    filteringCriteria   [1] FilteringCriteria { b2},
    extensions          [2] Extensions {b1}              OPTIONAL,
    responseCondition   [3] ResponseCondition            OPTIONAL,
    ...
    }

 setServiceProfile {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT        SetServiceProfileArg {b1,b2}
    RETURN RESULT   FALSE
    ERRORS          {missingParameter |
              parameterOutOfRange |
              taskRefused |
              unexpectedComponentSequence |
              unexpectedDataValue |
              unexpectedParameter }
    ALWAYS RESPONDS FALSE
    CODE    opcode-setServiceProfile
    }
-- Direction SCF -> SSF, Timer Tsep
-- This operation is used within the context of a call to request the SSF to
-- activate/de-activate  a list of trigger for one of the parties in the call.

SetServiceProfileArg {B1: b1, B2: b2}::= SEQUENCE {
        iNprofiles  [0]  SEQUENCE SIZE (1..b2.&numOfINProfile)
                         OF INprofile {b1, b2},
        ...,
        ...,
        extensions [30] Extensions {b1}        OPTIONAL
        }

splitLeg {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    SplitLegArg { b1, b2}
    RETURN RESULT TRUE
    ERRORS  {missingParameter |
            unexpectedComponentSequence |
            unexpectedParameter |
            unexpectedDataValue |
            systemFailure |
            taskRefused |
            unknownLegID
            }
    CODE    opcode-splitLeg
    }
-- Direction: SCF -> SSF. Timer: T_sl
-- This operation  is issued by the SCF  to separate one joined leg from a multi-way connection
-- or  a single 2 party Call segment.

SplitLegArg {B1: b1, B2: b2}::= SEQUENCE {
    legToBeSplit       [0] LegID,
    newCallSegment     [1] INTEGER (2..b2.&numOfCSs),
    extensions         [2] Extensions {b1}              OPTIONAL,
    ...
    }

statusReport {B1: b1, B2: b2}  OPERATION::= {
    ARGUMENT    StatusReportArg { b1, b2}
    RETURN RESULT    FALSE
    ALWAYS RESPONDS FALSE
    CODE    opcode-statusReport
    }
-- Direction: SSF ->  SCF, Timer: T_srp
-- This operation is used as a response to RequestFirstStatusMatchReport or
-- RequestEveryStatusChangeReport operations.
```

```
StatusReportArg {B1: b1, B2: b2}::= SEQUENCE {
    resourceStatus      [0] ResourceStatus              OPTIONAL,
    correlationID       [1] CorrelationID { b2}         OPTIONAL,
    resourceID          [2] ResourceID { b2}            OPTIONAL,
    extensions          [3] Extensions {b1}             OPTIONAL,
    reportCondition     [4] ReportCondition             OPTIONAL,
    ...
    }
-- For correlationID, OPTIONAL denotes network operator optional.
-- resourceID is required when the SSF sends a report as an answer to a previous request when the
-- correlationID was present.

END
```

14.3.1 Operation Timers

The following value ranges do apply for operation specific timers in INAP:
short: 1 - 10 seconds
medium: 1 - 60 seconds
long: 1 second - 30 minutes

Table 40 lists all operation timers and the value range for each timer. The definitive value for
each operation timer may be network specific and has to be defined by the network operator.

Table 40: Operation timers and their value range

Operation Name	Timer	value range
ActivateServiceFiltering	Tasf	medium
ActivityTest	T_{at}	short
ApplyCharging	T_{ac}	short
ApplyChargingReport	Tacr	short
AssistRequestInstructions	Tari	short
CallGap	T_{cg}	short
CallInformationReport	Tcirp	short
CallInformationRequest	Tcirq	short
Cancel	Tcan	short
CancelStatusReportRequest	Tcsr	short
CollectInformation	T_{ci}	medium
Connect	Tcon	short
ConnectToResource	Tctr	short
Continue	Tcue	short
ContinueWithArgument	Tcwa	short
CreateCallSegmentAssociation	Tcsa	short
CreateOrRemoveTriggerData	Tcrt	medium
DisconnectForwardConnection	Tdfc	short
DisconnectForwardConnectionWithArgument	Tdfcwa	short
DisconnectLeg	T_{dl}	short
EntityRelease	T_{er}	short
EstablishTemporaryConnection	Tetc	medium
EventNotificationCharging	Tenc	short
EventReportBCSM	Terb	short
FurnishChargingInformation	Tfci	short
InitialDP	Tidp	short
InitiateCallAttempt	Tica	short
ManageTriggerData	Tmtd	medium
MergeCallSegments	T_{mc}	short
MoveCallSegments	Tmcs	short
MoveLeg	T_{ml}	short
ReleaseCall	T_{rc}	short
ReportUTSI	T_{ru}	short
RequestCurrentStatusReport	Trcs	short
RequestEveryStatusChangeReport	Tres	short
RequestFirstStatusMatchReport	Trfs	short
RequestNotificationChargingEvent	Trnc	short
RequestReportBCSMEvent	Trrb	short
RequestReportUTSI	Trru	short
ResetTimer	T_{rt}	short
SelectFacility	T_{sf}	short
SendChargingInformation	Tsci	short
SendSTUI	T_{ss}	short
ServiceFilteringResponse	Tsfr	short
SetServiceProfile	Tsep	short
SplitLeg	T_{sl}	short
StatusReport	Tsrp	short

14.4 Packages, contracts, application contexts and abstract syntaxes

14.4.1 ASN.1 Modules

```
IN-CS3-SSF-SCF-pkgs-contracts-acs {itu-t(0) identified-organization(4) etsi(0) inDomain(1) in-
network(1) cs3(30) modules(1) in-cs3-ssf-scf-pkgs-contracts-acs(9) version1(0)}


DEFINITIONS::=

BEGIN

-- This module describes the operation-packages, contracts and application-contexts used
--  over the SSF-SCF interface.

IMPORTS

    IMSI,
    ISDN-AddressString,
    Ext-BasicServiceCode
FROM MAP-CommonDataTypes {ITU-T(0) identified-organization(4) etsi(0) mobileDomain(0) gsm-Network(1)
modules(3) map-CommonDataTypes(18) version6(6)}

    LocationInformation,
    SubscriberState
FROM MAP-MS-DataTypes {ITU-T(0) identified-organization(4) etsi(0) mobileDomain(0) gsm-Network(1)
modules(3) map-MS-DataTypes(11) version6(6)}

    CallReferenceNumber,
    SuppressionOfAnnouncement
FROM MAP-CH-DataTypes {ITU-T(0) identified-organization(4) etsi(0) mobileDomain(0) gsm-Network(1)
modules(3) map-CH-DataTypes(13) version6(6)}


    id-ac-cs3-ssf-scfGenericAC,
    id-ac-cs3-ssf-scfAssistHandoffAC,
    id-ac-cs3-ssf-scfServiceManagementAC,
    id-ac-cs3-scf-ssfGenericAC,
    id-ac-cs3-scf-ssfTrafficManagementAC,
    id-ac-cs3-scf-ssfServiceManagementAC,
    id-ac-cs3-scf-ssfStatusReportingAC,
    id-ac-cs3-scf-ssfTriggerManagementAC,

    id-inCs3SsfToScfGeneric,
    id-inCs3AssistHandoffSsfToScf,
    id-inCs3ScfToSsfGeneric,
    id-inCs3ScfToSsfTrafficManagement,
    id-inCs3ScfToSsfServiceManagement,
    id-inCs3SsfToScfServiceManagement,
    id-inCs3ScfToSsfStatusReporting,
    id-inCs3ScfToSsfTriggerManagement,

    id-as-ssf-scfGenericAS,
    id-as-assistHandoff-ssf-scfAS,
    id-as-scf-ssfGenericAS,
    id-as-scf-ssfTrafficManagementAS,
    id-as-scf-ssfServiceManagementAS,
    id-as-ssf-scfServiceManagementAS,
    id-as-scf-ssfStatusReportingAS,
    id-as-scf-ssfTriggerManagementAS,
    id-package-scfActivation,
    id-package-srf-scfActivationOfAssist,
    id-package-assistConnectionEstablishment,
    id-package-genericDisconnectResource,
    id-package-nonAssistedConnectionEstablishment,
    id-package-connect,
    id-package-callHandling,
    id-package-bcsmEventHandling,
    id-package-chargingEventHandling,
    id-package-ssfCallProcessing,
    id-package-scfCallInitiation,
    id-package-timer,
    id-package-billing,
```

```
        id-package-charging,
        id-package-trafficManagement,
        id-package-serviceManagementActivate,
        id-package-serviceManagementResponse,
        id-package-callReport,
        id-package-signallingControl,
        id-package-activityTest,
        id-package-statusReporting,
        id-package-cancel,
        id-package-cphResponse,
        id-package-entityReleased,
        id-package-triggerManagement,
        id-package-uSIHandling,
            id-package-triggerCallManagement,

    ros-InformationObjects,
    common-classes,
    scf-srf-classes,
    scf-srf-Operations,
    scf-srf-Protocol,
    ssf-scf-classes,
    ssf-scf-Operations,
    tc-Messages,
    tc-NotationExtensions
FROM IN-CS3-object-identifiers
    {itu-t(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) cs3(30) modules(1) in-
cs3-object-identifiers(0) version1(0)}

    COMMON-BOUNDS,
        EmptyReturnable
FROM IN-CS3-Common-Classes common-classes

    SCF-SSF-BOUNDS

FROM IN-CS3-SSF-SCF-Classes ssf-scf-classes

    SCF-SRF-BOUNDS

FROM IN-CS3-scf-srf-classes scf-srf-classes

    CONTRACT,
    OPERATION-PACKAGE,
    OPERATION,
    ROS-OBJECT-CLASS

FROM Remote-Operations-Information-Objects ros-InformationObjects

    TCMessage {}

FROM TCAPMessages tc-Messages

    APPLICATION-CONTEXT, dialogue-abstract-syntax

FROM TC-Notation-Extensions tc-NotationExtensions

    activateServiceFiltering {},
    activityTest,
    applyCharging {},
    applyChargingReport {},
    assistRequestInstructions {},
    callGap {},
    callInformationReport {},
    callInformationRequest {},
    cancel {},
    cancelStatusReportRequest {},
    collectInformation {},
    connect {},
    connectToResource {},
    continue,
    continueWithArgument {},
    createCallSegmentAssociation {},
    createOrRemoveTriggerData {},
    disconnectForwardConnection,
    disconnectForwardConnectionWithArgument {},
    disconnectLeg {},
    entityReleased {},
    establishTemporaryConnection {},
    eventNotificationCharging {},
```

```
        eventReportBCSM {},
        furnishChargingInformation {},
        initialDP {},
        initiateCallAttempt {},
        manageTriggerData {},
        mergeCallSegments {},
        moveCallSegments {},
        moveLeg {},
        releaseCall {},
        reportUTSI {},
        requestCurrentStatusReport {},
        requestEveryStatusChangeReport {},
        requestFirstStatusMatchReport {},
        requestNotificationChargingEvent {},
        requestReportBCSMEvent {},
        requestReportUTSI {},
        resetTimer {},
        selectFacility {},
        sendChargingInformation {},
        sendSTUI {},
        serviceFilteringResponse {},
        setServiceProfile {},
        splitLeg {},
        statusReport {}

FROM IN-CS3-SSF-SCF-ops-args ssf-scf-Operations

        playAnnouncement {},
        promptAndCollectUserInformation {},
        promptAndReceiveMessage {},
        scriptClose {},
        scriptEvent {},
        scriptInformation {},
        scriptRun {},
        specializedResourceReport

FROM IN-CS3-scf-srf-ops-args scf-srf-Operations

        specializedResourceControlPackage {},
        scriptControlPackage {},
        messageControlPackage {}

FROM IN-CS3-SCF-SRF-pkgs-contracts-acs scf-srf-Protocol
;
-- The following three definitions are local short-hand notation for convenience.
B1::= COMMON-BOUNDS      -- defined in EN 301 931-1
B2::= SCF-SSF-BOUNDS        -- defined in this part
B3::= SCF-SRF-BOUNDS        -- defined in EN 301 931-3
networkSpecificB1  COMMON-BOUNDS::=
{   NUM-OF-EXTENSIONS   1}


-- The following instance of the parameter bound is just an example

networkSpecificB2  SCF-SSF-BOUNDS::=
{
        MINIMUM-FOR-ACH-BILLING-CHARGING         1  -- example value
        MAXIMUM-FOR-ACH-BILLING-CHARGING         5  -- example value
        MINIMUM-FOR-BACKWARD-GVNS                1  -- example value
        MAXIMUM-FOR-BACKWARD-GVNS                5  -- example value
        MAXIMUM-FOR-BEARER-CAPABILITY            5  -- example value
        MINIMUM-FOR-CALLED-DIRECTORY-NUMBER      1  -- example value
        MAXIMUM-FOR-CALLED-DIRECTORY-NUMBER      5  -- example value
        MINIMUM-FOR-CALLED-PARTY-BCD-NUMBER      1  -- example value
        MAXIMUM-FOR-CALLED-PARTY-BCD-NUMBER      5  -- example value
        MINIMUM-FOR-CALLED-PARTY-NUMBER          1  -- example value
        MAXIMUM-FOR-CALLED-PARTY-NUMBER          5  -- example value
        MINIMUM-FOR-CALLING-GEODETIC-LOCATION    1  -- example value
        MAXIMUM-FOR-CALLING-GEODETIC-LOCATION    10 -- example value
        MINIMUM-FOR-CALLING-PARTY-NUMBER         1  -- example value
        MAXIMUM-FOR-CALLING-PARTY-NUMBER         5  -- example value
        MINIMUM-FOR-CALL-RESULT-CSONE            1  -- example value
        MAXIMUM-FOR-CALL-RESULT-CSONE            5  -- example value
        MAXIMUM-FOR-CALL-REFERENCE               5  -- example value
        MINIMUM-FOR-CARRIER                      3  -- example value
        MAXIMUM-FOR-CARRIER                      10 -- example value
        MINIMUM-FOR-CHARGED-PARTYNUMBER          1  -- example value
        MAXIMUM-FOR-CHARGED-PARTYNUMBER          14 -- example value
        MAXIMUM-FOR-CAUSE                        4  -- example value
```

```
MINIMUM-FOR-COMMUNICATION-TARIFF-NUM              1    -- example value
MAXIMUM-FOR-COMMUNICATION-TARIFF-NUM              4    -- example value
MAXIMUM-FOR-CNINFO                                13   -- example value
MINIMUM-FOR-DIGITS                                1    -- example value
MAXIMUM-FOR-DIGITS                                5    -- example value
MINIMUM-FOR-DISPLAY                               1    -- example value
MAXIMUM-FOR-DISPLAY                               5    -- example value
MINIMUM-FOR-EVENT-SPECIFIC-CHARGING               1    -- example value
MAXIMUM-FOR-EVENT-SPECIFIC-CHARGING               5    -- example value
MINIMUM-FOR-EVENT-TYPE-CHARGING                   1    -- example value
MAXIMUM-FOR-EVENT-TYPE-CHARGING                   5    -- example value
MINIMUM-FOR-FCI-CSONE-BILLING-CHARGING            1    -- example value
MAXIMUM-FOR-FCI-CSONE-BILLING-CHARGING            5    -- example value
MINIMUM-FOR-FCI-CSTWO-BILLING-CHARGING            1    -- example value
MAXIMUM-FOR-FCI-CSTWO-BILLING-CHARGING            5    -- example value
MINIMUM-FOR-FCI-BILLING-CHARGING-DATA             1    -- example value
MAXIMUM-FOR-FCI-BILLING-CHARGING-DATA             160  -- example value
MINIMUM-FOR-CAMELFCI-BILLING-CHARGING-DATA        1    -- example value
MAXIMUM-FOR-CAMELFCI-BILLING-CHARGING-DATA        5    -- example value
MINIMUM-FOR-FCI-BILLING-CHARGING                  5    -- example value
MAXIMUM-FOR-FCI-BILLING-CHARGING                  172  -- example value
MINIMUM-FOR-FORMAT-DATA-LENGTH                    1    -- example value
MAXIMUM-FOR-FORMAT-DATA-LENGTH                    5    -- example value
MINIMUM-FOR-FORWARD-GVNS                          1    -- example value
MAXIMUM-FOR-FORWARD-GVNS                          5    -- example value
MINIMUM-FOR-GENERIC-NAME                          1    -- example value
MAXIMUM-FOR-GENERIC-NAME                          5    -- example value
MINIMUM-FOR-GENERIC-NUMBER                        1    -- example value
MAXIMUM-FOR-GENERIC-NUMBER                        5    -- example value
MAXIMUM-FOR-GLOBAL-CALLREF                        10   -- example value
MAXIMUM-FOR-INITIAL-TIME-INTERVAL                 5    -- example value
MAXIMUM-FOR-IN-SERVICE-COMPATIBILITY              5    -- example value
MINIMUM-FOR-IP-AVAILABLE                          1    -- example value
MAXIMUM-FOR-IP-AVAILABLE                          5    -- example value
MINIMUM-FOR-IP-SSP-CAPABILITIES                   1    -- example value
MAXIMUM-FOR-IP-SSP-CAPABILITIES                   5    -- example value
MINIMUM-FOR-ISDN-ACCESS-RELATED-INFO              1    -- example value
MAXIMUM-FOR-ISDN-ACCESS-RELATED-INFO              10   -- example value
MINIMUM-FOR-LOCATION-NUMBER                       1    -- example value
MAXIMUM-FOR-LOCATION-NUMBER                       5    -- example value
MINIMUM-FOR-MID-CALL-CONTROL-INFO                 1    -- example value
MAXIMUM-FOR-MID-CALL-CONTROL-INFO                 5    -- example value
MINIMUM-FOR-ORIGINAL-CALLED-PARTY-ID              1    -- example value
MAXIMUM-FOR-ORIGINAL-CALLED-PARTY-ID              5    -- example value
MINIMUM-FOR-REASON                                1    -- example value
MAXIMUM-FOR-REASON                                5    -- example value
MINIMUM-FOR-REDIRECTING-ID                        1    -- example value
MAXIMUM-FOR-REDIRECTING-ID                        5    -- example value
MINIMUM-FOR-REQUESTED-UTSI-NUM                    1    -- example value
MAXIMUM-FOR-REQUESTED-UTSI-NUM                    5    -- example value
MINIMUM-FOR-ROUTE-LIST                            1    -- example value
MAXIMUM-FOR-ROUTE-LIST                            5    -- example value
MINIMUM-FOR-ROUTING-NUMBER                        1    -- example value
MAXIMUM-FOR-ROUTING-NUMBER                        5    -- example value
MINIMUM-FOR-SCF-ID                                1    -- example value
MAXIMUM-FOR-SCF-ID                                5    -- example value
MINIMUM-FOR-SCI-BILLING-CHARGING                  1    -- example value
MAXIMUM-FOR-SCI-BILLING-CHARGING                  5    -- example value
MINIMUM-FOR-SDSS-INFORMATION                      1    -- example value
MAXIMUM-FOR-SDSS-INFORMATION                      5    -- example value
MINIMUM-FOR-SII                                   1    -- example value
MAXIMUM-FOR-SII                                   5    -- example value
MINIMUM-FOR-SF-BILLING-CHARGING                   1    -- example value
MAXIMUM-FOR-SF-BILLING-CHARGING                   5    -- example value
MINIMUM-FOR-SUB-TARRIF-CONTROL-LENGTH             1    -- example value
MAXIMUM-FOR-SUB-TARRIF-CONTROL-LENGTH             5    -- example value
MINIMUM-FOR-TIME-AND-TIMEZONE-LENGTH              1    -- example value
MAXIMUM-FOR-TIME-AND-TIMEZONE-LENGTH              8    -- example value
MINIMUM-FOR-TARIFF-INDICATORS-LENGTH              1    -- example value
MAXIMUM-FOR-TARIFF-INDICATORS-LENGTH              8    -- example value
MINIMUM-FOR-USI-INFORMATION                       1    -- example value
MAXIMUM-FOR-USI-INFORMATION                       5    -- example value
MINIMUM-FOR-USI-SERVICE-INDICATOR                 1    -- example value
MAXIMUM-FOR-USI-SERVICE-INDICATOR                 5    -- example value
NUM-OF-BCSM-EVENT                                 4    -- example value
NUM-OF-BCUSM-EVENT                                4    -- example value
NUM-OF-CHARGING-EVENT                             4    -- example value
NUM-OF-CSAS                                       2    -- example value
```

```
        NUM-OF-CSS                                      2    -- example value
        NUM-OF-GENERIC-NUMBERS                          2    -- example value
        NUM-OF-INPROFILE                                1    -- example value
        NUM-OF-SEVERALTRIGGER                           5    -- example value
        NUM-OF-IN-SERVICE-COMPATIBILITY-ID              2    -- example value
        NUM-OF-LEGS                                     2    -- example value
        NUM-OF-MESSAGE-IDS                              2    -- example value
        MAXIMUM-FOR-AMOUNT                              2    -- example value
        MAXIMUM-FOR-INITIAL-UNIT-INCREMENT             2    -- example value
        MAXIMUM-FOR-SCALING-FACTOR                      2    -- example value
        MAXIMUM-FOR-SEGMENTS-PER-DATA-INTERVAL          5    -- example value
        MAXIMUM-FOR-UB-NB-CALL                          5    -- example value
        NUM-OF-ADDRESSES                                5    -- example value
}

networkSpecificB3  SCF-SRF-BOUNDS::=
{
        MINIMUM-FOR-ATTRIBUTES                          1    -- example value
        MAXIMUM-FOR-ATTRIBUTES                          5    -- example value
        MINIMUM-FOR-MAIL-BOX-ID                         1    -- example value
        MAXIMUM-FOR-MAIL-BOX-ID                         5    -- example value
        MINIMUM-FOR-MESSAGE-CONTENT                     1    -- example value
        MAXIMUM-FOR-MESSAGE-CONTENT                     5    -- example value
        MINIMUM-FOR-RECEIVED-INFORMATION                1    -- example value
        MAXIMUM-FOR-RECEIVED-INFORMATION                5    -- example value
        MAXIMUM-FOR-RECORDING-TIME                      5    -- example value
        NUM-OF-MESSAGE-IDS                              2    -- example value
        MAXIMUM-FOR-RECORDED-MESSAGE-UNITS              5    -- example value
        NUM-OF-VARIABLE-PARTS                           5    -- must be  5 or
  --      greater.
}



-- Application Contexts --


cs3ssf-scfGenericAC  APPLICATION-CONTEXT::= {
    CONTRACT      inCs3SsfToScfGeneric
    DIALOGUE MODE     structured
    ABSTRACT SYNTAXES    {dialogue-abstract-syntax |
            ssf-scfGenericAbstractSyntax}
    APPLICATION CONTEXT NAME     id-ac-cs3-ssf-scfGenericAC}


cs3ssf-scfAssistHandoffAC  APPLICATION-CONTEXT::=   {
    CONTRACT      inCs3AssistHandoffSsfToScf
    DIALOGUE MODE     structured
    ABSTRACT SYNTAXES    {dialogue-abstract-syntax |
            assistHandoff-ssf-scfAbstractSyntax}
    APPLICATION CONTEXT NAME     id-ac-cs3-ssf-scfAssistHandoffAC}

cs3ssf-scfServiceManagementAC  APPLICATION-CONTEXT::= {
    CONTRACT      inCs3SsfToScfServiceManagement
    DIALOGUE MODE     structured
    ABSTRACT SYNTAXES    {dialogue-abstract-syntax |
            ssf-scfServiceManagementAbstractSyntax}
    APPLICATION CONTEXT NAME     id-ac-cs3-ssf-scfServiceManagementAC}


cs3scf-ssfGenericAC  APPLICATION-CONTEXT::= {
    CONTRACT      inCs3ScfToSsfGeneric
    DIALOGUE MODE     structured
    ABSTRACT SYNTAXES    {dialogue-abstract-syntax |
            ssf-scfGenericAbstractSyntax}
    APPLICATION CONTEXT NAME     id-ac-cs3-scf-ssfGenericAC}

cs3scf-ssfTrafficManagementAC  APPLICATION-CONTEXT::= {
    CONTRACT      inCs3ScfToSsfTrafficManagement
    DIALOGUE MODE     structured
    ABSTRACT SYNTAXES    {dialogue-abstract-syntax |
            scf-ssfTrafficManagementAbstractSyntax}
    APPLICATION CONTEXT NAME     id-ac-cs3-scf-ssfTrafficManagementAC}
```

```
cs3scf-ssfServiceManagementAC  APPLICATION-CONTEXT::= {
    CONTRACT    inCs3ScfToSsfServiceManagement
    DIALOGUE MODE    structured
    ABSTRACT SYNTAXES    {dialogue-abstract-syntax |
            scf-ssfServiceManagementAbstractSyntax}
    APPLICATION CONTEXT NAME    id-ac-cs3-scf-ssfServiceManagementAC}

cs3scf-ssfStatusReportingAC  APPLICATION-CONTEXT::= {
    CONTRACT    inCs3ScfToSsfStatusReporting
    DIALOGUE MODE    structured
    ABSTRACT SYNTAXES    {dialogue-abstract-syntax |
            scf-ssfStatusReportingAbstractSyntax}
    APPLICATION CONTEXT NAME    id-ac-cs3-scf-ssfStatusReportingAC}

cs3scf-ssfTriggerManagementAC  APPLICATION-CONTEXT::= {
    CONTRACT    inCs3ScfToSsfTriggerManagement
    DIALOGUE MODE    structured
    ABSTRACT SYNTAXES    {dialogue-abstract-syntax |
            scf-ssfTriggerManagementAbstractSyntax}
    APPLICATION CONTEXT NAME    id-ac-cs3-scf-ssfTriggerManagementAC}



-- Contracts --


inCs3SsfToScfGeneric CONTRACT::= {
-- dialogue initiated by SSF with InitialDP Operation
--The inCs3SsfToScfGeneric contract expresses the form of the service in which the SSF,
--a ROS object of class ssf, initiates the generic triggering approach contract.
--A ROS-object of class scf, responds to this contract.

    INITIATOR CONSUMER OF   { activityTestPackage|
            exceptionInformPackage  {networkSpecificB2} |
            scfActivationPackage {networkSpecificB1, networkSpecificB2} }
    RESPONDER CONSUMER OF    {activityTestPackage|
            assistConnectionEstablishmentPackage {networkSpecificB1, networkSpecificB2} |
            bcsmEventHandlingPackage {networkSpecificB1, networkSpecificB2} |
            billingPackage {networkSpecificB1, networkSpecificB2} |
            callHandlingPackage { networkSpecificB2} |
            callReportPackage {networkSpecificB1, networkSpecificB2} |
            cancelPackage {networkSpecificB2} |
            chargingEventHandlingPackage {networkSpecificB1, networkSpecificB2} |
            chargingPackage {networkSpecificB1, networkSpecificB2} |
            connectPackage {networkSpecificB1, networkSpecificB2} |
            cphResponsePackage  {networkSpecificB1, networkSpecificB2} |
            genericDisconnectResourcePackage {networkSpecificB1, networkSpecificB2} |
            nonAssistedConnectionEstablishmentPackage

{networkSpecificB1, networkSpecificB2} |
            signallingControlPackage {networkSpecificB1, networkSpecificB2} |
            specializedResourceControlPackage {networkSpecificB1, networkSpecificB2,
networkSpecificB3} |
            scriptControlPackage {networkSpecificB1, networkSpecificB2} |
            messageControlPackage {networkSpecificB1, networkSpecificB2, networkSpecificB3} |
            ssfCallProcessingPackage {networkSpecificB1 } |
            statusReportingPackage {networkSpecificB1, networkSpecificB2} |
            timerPackage {networkSpecificB1, networkSpecificB2} |
            trafficManagementPackage {networkSpecificB1, networkSpecificB2, networkSpecificB3} |
            uSIHandlingPackage {networkSpecificB1, networkSpecificB2} |
            scfCallInitiationPackage {networkSpecificB1, networkSpecificB2}|
            triggerCallManagementPackage {networkSpecificB1, networkSpecificB2}
            }
    ID      id-inCs3SsfToScfGeneric

    }


inCs3AssistHandoffSsfToScf CONTRACT::= {
-- dialogue initiated by SSF with AssistRequestInstructions
--The inCs3AssistHandoffSsfToScf contract expresses the form of the service in which the SSF,
-- a ROS-object of class ssf, initiates the Assist or Hand-off contract.
--A ROS-object of class scf, responds to this contract.
```

```
    INITIATOR CONSUMER OF    { activityTestPackage|
            srf-scfActivationOfAssistPackage {networkSpecificB1, networkSpecificB2} }
    RESPONDER CONSUMER OF    {activityTestPackage|
            billingPackage {networkSpecificB1, networkSpecificB2} |
            callHandlingPackage { networkSpecificB2} |
            cancelPackage {networkSpecificB2} |
            chargingPackage {networkSpecificB1, networkSpecificB2} |
            signallingControlPackage {networkSpecificB1, networkSpecificB2} |
            genericDisconnectResourcePackage {networkSpecificB1, networkSpecificB2} |
            nonAssistedConnectionEstablishmentPackage {networkSpecificB1, networkSpecificB2} |
            specializedResourceControlPackage {networkSpecificB1, networkSpecificB2,
            networkSpecificB3} |
            scriptControlPackage {networkSpecificB1, networkSpecificB2} |
            messageControlPackage {networkSpecificB1, networkSpecificB2, networkSpecificB3} |
            statusReportingPackage {networkSpecificB1, networkSpecificB2} |
            timerPackage {networkSpecificB1, networkSpecificB2}
            }
    ID      id-inCs3AssistHandoffSsfToScf
    }

inCs3ScfToSsfGeneric CONTRACT::= {
--dialogue initiated by SCF with InitiateCallAttempt or CreateCallSegmentAssociation, Generic Case
--The inCs3ScfToSsfGeneric contract expresses the form of the service in which the SCF,
--a ROS-object of class scf, initiates the generic messaging approach for the SCF
--Initiate Call Attempt contract. A ROS-object of class ssf, responds to this contract.

    INITIATOR CONSUMER OF    {activityTestPackage|
            assistConnectionEstablishmentPackage {networkSpecificB1, networkSpecificB2} |
            bcsmEventHandlingPackage {networkSpecificB1, networkSpecificB2} |
            billingPackage {networkSpecificB1, networkSpecificB2} |
            callHandlingPackage { networkSpecificB2} |
            callReportPackage {networkSpecificB1, networkSpecificB2} |
            cancelPackage { networkSpecificB2} |
            chargingEventHandlingPackage {networkSpecificB1, networkSpecificB2} |
            chargingPackage {networkSpecificB1, networkSpecificB2} |
            connectPackage {networkSpecificB1, networkSpecificB2} |
            cphResponsePackage  {networkSpecificB1, networkSpecificB2} |
            genericDisconnectResourcePackage {networkSpecificB1, networkSpecificB2} |
            nonAssistedConnectionEstablishmentPackage {networkSpecificB1, networkSpecificB2} |
            scfCallInitiationPackage {networkSpecificB1, networkSpecificB2} |
            signallingControlPackage {networkSpecificB1, networkSpecificB2} |
            specializedResourceControlPackage {networkSpecificB1, networkSpecificB2,
            networkSpecificB3} |
            scriptControlPackage {networkSpecificB1, networkSpecificB2} |
            messageControlPackage {networkSpecificB1, networkSpecificB2, networkSpecificB3} |
            ssfCallProcessingPackage {networkSpecificB1 } |
            statusReportingPackage {networkSpecificB1, networkSpecificB2} |
            timerPackage {networkSpecificB1, networkSpecificB2} |
            uSIHandlingPackage {networkSpecificB1, networkSpecificB2} |
            scfCallInitiationPackage {networkSpecificB1, networkSpecificB2} |
            triggerCallManagementPackage {networkSpecificB1, networkSpecificB2}
            }

    RESPONDER CONSUMER OF    { activityTestPackage|
     exceptionInformPackage {networkSpecificB2} }
    ID      id-inCs3ScfToSsfGeneric
    }


inCs3ScfToSsfTrafficManagement CONTRACT::= {
-- dialogue initiated by SCF with CallGap
-- The inCs3ScfToSsfTrafficManagement contract expresses the form of the service in
-- which the SCF, a ROS-object of class scf, initiates the Traffic Management related contract.
-- A ROS-object of class ssf, responds to this contract.

    INITIATOR CONSUMER OF    {trafficManagementPackage {networkSpecificB1, networkSpecificB2,
networkSpecificB3}
            }
    ID      id-inCs3ScfToSsfTrafficManagement
    }
inCs3ScfToSsfServiceManagement CONTRACT::= {
-- dialogue initiated by SCF with ActivateServiceFiltering
--The inCs3ScfToSsfServiceManagement contract expresses the form of the service
-- in which the SCF, a ROS-object of class scf, initiates the Service Management related contract.
-- A ROS-object of class ssf, in the context of a separate contract, responds to this initiation.
```

```
        INITIATOR CONSUMER OF   {serviceManagementActivatePackage {networkSpecificB1, networkSpecificB2,
            networkSpecificB3}
            }
        ID      id-inCs3ScfToSsfServiceManagement
    }
inCs3SsfToScfServiceManagement CONTRACT::= {
-- dialogue initiated/ended by SSF with ServiceFilteringResponse
-- The inCs3SsfToScfServiceManagement contract expresses the form of the service in
-- which the SSF, a ROS-object of class ssf, initiates the Service Management related contract
-- for reporting Service Management results.

        INITIATOR CONSUMER OF   {serviceManagementResponsePackage {networkSpecificB1, networkSpecificB2}
            }
        ID      id-inCs3SsfToScfServiceManagement
    }
inCs3ScfToSsfStatusReporting CONTRACT::= {
-- dialogue initiated by SCF with StatusReporting Operations
--The inCs3ScfToSsfStatusReporting contract expresses the form of the service
--in which the SCF, a ROS-object of class scf, initiates the Status Reporting related contract.
--A ROS-object of class ssf, responds to this contract.

        INITIATOR CONSUMER OF   {
            statusReportingPackage {networkSpecificB1, networkSpecificB2}
            }
        ID      id-inCs3ScfToSsfStatusReporting
    }

inCs3ScfToSsfTriggerManagement CONTRACT::= {
-- dialogue initiated by SCF with the trigger management operations CreateOrRemoveTriggerData
-- or ManageTriggerData
-- The inCs3ScfToSsfTriggerManagement contract expresses the form of the service in which the SCF,
-- a ROS-object of class scf, initiates the Trigger Management related contract
-- A ROS-object of class ssf, in the context of a separate contract, responds to this initiation.
        INITIATOR CONSUMER OF   {triggerManagementPackage {networkSpecificB1, networkSpecificB2,
networkSpecificB3}
            }
        ID      id-inCs3ScfToSsfTriggerManagement
    }


-- Packages --


scfActivationPackage  {B1: b1, B2: b2}   OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            initialDP { b1, b2}
            }
    ID      id-package-scfActivation}


srf-scfActivationOfAssistPackage  {B1: b1, B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            assistRequestInstructions {b1, b2}
            }
    ID      id-package-srf-scfActivationOfAssist}

assistConnectionEstablishmentPackage  { B1: b1, B2: b2}     OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            establishTemporaryConnection {b1,b2}
            }
    ID      id-package-assistConnectionEstablishment}

genericDisconnectResourcePackage  { B1: b1, B2: b2}     OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            disconnectForwardConnection |
            disconnectForwardConnectionWithArgument {b1, b2}
            }
    ID      id-package-genericDisconnectResource}

nonAssistedConnectionEstablishmentPackage  {B1: b1, B2: b2}  OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            connectToResource {b1, b2}
            }
    ID      id-package-nonAssistedConnectionEstablishment}
```

```
connectPackage  {B1: b1, B2: b2}  OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            connect {b1, b2}
            }
    ID      id-package-connect}


callHandlingPackage   {B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            releaseCall {b2}
            }
    ID      id-package-callHandling}


bcsmEventHandlingPackage  {B1: b1, B2: b2}  OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            requestReportBCSMEvent {b1, b2}
            }
    SUPPLIER INVOKES    {
            eventReportBCSM {b1, b2}
            }
    ID      id-package-bcsmEventHandling}



chargingEventHandlingPackage  {B1: b1, B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            requestNotificationChargingEvent {b2}
            }
    SUPPLIER INVOKES    {
            eventNotificationCharging {b1, b2}
            }
    ID      id-package-chargingEventHandling}

ssfCallProcessingPackage {B1: b1}  OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            collectInformation  {b1} |
            selectFacility  {b1}|
            continue
            }
    ID      id-package-ssfCallProcessing}

scfCallInitiationPackage {B1: b1, B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            initiateCallAttempt {b1, b2}
            }
    ID      id-package-scfCallInitiation}

timerPackage {B1: b1, B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            resetTimer {b1, b2}
            }
    ID      id-package-timer}

billingPackage {B1: b1, B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            furnishChargingInformation {b1, b2}
            }
    ID      id-package-billing}

chargingPackage {B1: b1, B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            applyCharging {b1, b2}
            }
    SUPPLIER INVOKES    {
            applyChargingReport {b1, b2}
            }
    ID      id-package-charging}

trafficManagementPackage {B1: b1, B2: b2, B3: b3 } OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            callGap {b1, b2, b3}
            }
    ID      id-package-trafficManagement}

serviceManagementActivatePackage {B1: b1, B2: b2, B3: b3} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            activateServiceFiltering {b1, b2, b3}
            }
    ID      id-package-serviceManagementActivate}
```

```
serviceManagementResponsePackage {B1: b1, B2: b2 } OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            serviceFilteringResponse {b1, b2}
            }
    ID      id-package-serviceManagementResponse}


callReportPackage {B1: b1, B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            callInformationRequest {b1}
            }
    SUPPLIER INVOKES    {
            callInformationReport {b1, b2}
            }
    ID      id-package-callReport}


signallingControlPackage {B1: b1, B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            sendChargingInformation {b1, b2}
            }
    ID      id-package-signallingControl}


activityTestPackage  OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            activityTest
            }
    ID      id-package-activityTest}


statusReportingPackage {B1: b1, B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            cancelStatusReportRequest {b1, b2}|
            requestCurrentStatusReport  {b1, b2}|
            requestEveryStatusChangeReport {b1, b2}|
            requestFirstStatusMatchReport {b1, b2}
            }
    SUPPLIER INVOKES    {
            statusReport {b1, b2}
            }
    ID      id-package-statusReporting}


cancelPackage {B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            cancel {b2}


            }
    ID      id-package-cancel}


cphResponsePackage  {B1: b1, B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            continueWithArgument {b1, b2}|
            disconnectLeg  {b1, b2}|
            mergeCallSegments  {b1, b2}|
            moveCallSegments  {b1, b2}|
            moveLeg  {b1, b2}|
            createCallSegmentAssociation  {b1, b2} |
            splitLeg {b1, b2}
            }
    ID      id-package-cphResponse}


exceptionInformPackage   {B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            entityReleased {b2}
            }
    ID      id-package-entityReleased}


triggerCallManagementPackage   {B1: b1, B2: b2} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            setServiceProfile {b1, b2}
            }
    ID      id-package-triggerCallManagement}


triggerManagementPackage    {B1: b1, B2: b2, B3:b3} OPERATION-PACKAGE::= {
    CONSUMER INVOKES    {
            createOrRemoveTriggerData {b1, b2, b3}|
            manageTriggerData {b1, b2}
            }
    ID      id-package-triggerManagement}


uSIHandlingPackage   {B1: b1, B2: b2} OPERATION-PACKAGE::= {
```

```
    CONSUMER INVOKES    {
            requestReportUTSI {b1, b2}|
            sendSTUI {b1, b2}}
    SUPPLIER INVOKES    {
            reportUTSI {b1, b2}
            }
    ID      id-package-uSIHandling
    }


-- Abstract Syntaxes --

ssf-scfGenericAbstractSyntax  ABSTRACT-SYNTAX::= {
    GenericSSF-SCF-PDUs
    IDENTIFIED BY   id-as-ssf-scfGenericAS}

GenericSSF-SCF-PDUs::= TCMessage    {{SsfToScfGenericInvokable},
            {SsfToScfGenericReturnable}}

SsfToScfGenericInvokable  OPERATION::= {
            activateServiceFiltering  {networkSpecificB1, networkSpecificB2, networkSpecificB3} |
            activityTest |
            applyCharging  {networkSpecificB1, networkSpecificB2} |
            applyChargingReport  {networkSpecificB1, networkSpecificB2} |
            callInformationReport  {networkSpecificB1, networkSpecificB2} |
            callInformationRequest  {networkSpecificB1} |
            cancel  {networkSpecificB2} |
            cancelStatusReportRequest  {networkSpecificB1, networkSpecificB2} |
            collectInformation  {networkSpecificB1 } |
            connect  {networkSpecificB1, networkSpecificB2} |
            connectToResource  {networkSpecificB1, networkSpecificB2}|
            createCallSegmentAssociation {networkSpecificB1, networkSpecificB2} |
            disconnectForwardConnection |
            disconnectForwardConnectionWithArgument {networkSpecificB1, networkSpecificB2}|
            disconnectLeg {networkSpecificB1, networkSpecificB2} |
            entityReleased {networkSpecificB2} |
            establishTemporaryConnection {networkSpecificB1, networkSpecificB2} |
            eventNotificationCharging  {networkSpecificB1, networkSpecificB2} |
            eventReportBCSM  {networkSpecificB1, networkSpecificB2} |
            furnishChargingInformation  {networkSpecificB1, networkSpecificB2} |
            initialDP  {networkSpecificB1, networkSpecificB2} |
            mergeCallSegments {networkSpecificB1, networkSpecificB2} |
            moveCallSegments {networkSpecificB1, networkSpecificB2} |
            moveLeg {networkSpecificB1, networkSpecificB2} |
            releaseCall { networkSpecificB2} |
            reportUTSI {networkSpecificB1, networkSpecificB2} |
            requestCurrentStatusReport  {networkSpecificB1, networkSpecificB2} |
            requestEveryStatusChangeReport {networkSpecificB1, networkSpecificB2} |
            requestFirstStatusMatchReport {networkSpecificB1, networkSpecificB2} |
            requestNotificationChargingEvent  {networkSpecificB2} |
            requestReportBCSMEvent  {networkSpecificB1, networkSpecificB2} |
            requestReportUTSI {networkSpecificB1, networkSpecificB2} |
            resetTimer  {networkSpecificB1, networkSpecificB2} |
            sendChargingInformation  {networkSpecificB1, networkSpecificB2} |
            sendSTUI {networkSpecificB1, networkSpecificB2} |
            serviceFilteringResponse  {networkSpecificB1, networkSpecificB2} |
            setServiceProfile {networkSpecificB1, networkSpecificB2}|
            splitLeg {networkSpecificB1, networkSpecificB2} |
            statusReport  {networkSpecificB1, networkSpecificB2} |
            playAnnouncement  {networkSpecificB1, networkSpecificB2, networkSpecificB3 } |
            promptAndCollectUserInformation  {networkSpecificB1, networkSpecificB2,
            networkSpecificB3 } |
            scriptClose  {networkSpecificB1, networkSpecificB2} |
            scriptEvent  {networkSpecificB1, networkSpecificB2} |
            scriptInformation  {networkSpecificB1, networkSpecificB2} |
            scriptRun  {networkSpecificB1, networkSpecificB2} |
            specializedResourceReport |
            promptAndReceiveMessage  {networkSpecificB1, networkSpecificB2, networkSpecificB3 }
            }

SsfToScfGenericReturnable  OPERATION::= {
            activateServiceFiltering  {networkSpecificB1, networkSpecificB2, networkSpecificB3} |
            activityTest |
            applyCharging  {networkSpecificB1, networkSpecificB2} |
            applyChargingReport  {networkSpecificB1, networkSpecificB2} |
            callGap  {networkSpecificB1, networkSpecificB2, networkSpecificB3} |
            callInformationRequest  {networkSpecificB1} |
            cancel  {networkSpecificB2} |
```

```
           cancelStatusReportRequest  {networkSpecificB1, networkSpecificB2} |
           collectInformation {networkSpecificB1 } |
           connect {networkSpecificB1, networkSpecificB2} |
           connectToResource {networkSpecificB1, networkSpecificB2} |
           continue |
           continueWithArgument  {networkSpecificB1, networkSpecificB2}|
           createCallSegmentAssociation {networkSpecificB1, networkSpecificB2}|
           disconnectForwardConnection |
           disconnectForwardConnectionWithArgument {networkSpecificB1, networkSpecificB2}|
           disconnectLeg {networkSpecificB1, networkSpecificB2}|
           establishTemporaryConnection {networkSpecificB1, networkSpecificB2}|
           furnishChargingInformation  {networkSpecificB1, networkSpecificB2}|
           initialDP  {networkSpecificB1, networkSpecificB2}|
           mergeCallSegments {networkSpecificB1, networkSpecificB2}|
           moveCallSegments {networkSpecificB1, networkSpecificB2}|
           moveLeg {networkSpecificB1, networkSpecificB2}|
           releaseCall { networkSpecificB2}|
           requestCurrentStatusReport  {networkSpecificB1, networkSpecificB2}|
           requestEveryStatusChangeReport {networkSpecificB1, networkSpecificB2}|
           requestFirstStatusMatchReport  {networkSpecificB1, networkSpecificB2}|
           requestNotificationChargingEvent  {networkSpecificB2}|
           requestReportBCSMEvent  {networkSpecificB1, networkSpecificB2}|
           requestReportUTSI {networkSpecificB1, networkSpecificB2}|
           resetTimer  {networkSpecificB1, networkSpecificB2}|
           sendChargingInformation {networkSpecificB1, networkSpecificB2}|
           sendSTUI {networkSpecificB1, networkSpecificB2}|
           setServiceProfile {networkSpecificB1, networkSpecificB2}|
           splitLeg {networkSpecificB1, networkSpecificB2}|
           playAnnouncement  {networkSpecificB1, networkSpecificB2, networkSpecificB3 }|
           promptAndCollectUserInformation  {networkSpecificB1, networkSpecificB2,
           networkSpecificB3 }|
           scriptClose  {networkSpecificB1, networkSpecificB2}|
           scriptInformation  {networkSpecificB1, networkSpecificB2}|
           scriptRun  {networkSpecificB1, networkSpecificB2}|
           promptAndReceiveMessage  {networkSpecificB1, networkSpecificB2, networkSpecificB3 }
           }


assistHandoff-ssf-scfAbstractSyntax  ABSTRACT-SYNTAX::= {
     AssistHandoffSSF-SCF-PDUs
     IDENTIFIED BY   id-as-assistHandoff-ssf-scfAS}

AssistHandoffSSF-SCF-PDUs::= TCMessage {{AssistHandoffSsfToScfInvokable},
           {AssistHandoffSsfToScfReturnable}}

AssistHandoffSsfToScfInvokable OPERATION::= {
           activityTest |
           applyCharging  {networkSpecificB1, networkSpecificB2}|
           applyChargingReport  {networkSpecificB1, networkSpecificB2}|
           assistRequestInstructions  {networkSpecificB1, networkSpecificB2}|
           cancel  {networkSpecificB2}|
           cancelStatusReportRequest  {networkSpecificB1, networkSpecificB2}|
           connectToResource  {networkSpecificB1, networkSpecificB2}|
           disconnectForwardConnection |
           disconnectForwardConnectionWithArgument {networkSpecificB1, networkSpecificB2}|
           furnishChargingInformation  {networkSpecificB1, networkSpecificB2}|
           playAnnouncement  {networkSpecificB1, networkSpecificB2, networkSpecificB3 }|
           promptAndCollectUserInformation  {networkSpecificB1, networkSpecificB2,
           networkSpecificB3 }|
           requestCurrentStatusReport  {networkSpecificB1, networkSpecificB2}|
           requestEveryStatusChangeReport {networkSpecificB1, networkSpecificB2}|
           requestFirstStatusMatchReport  {networkSpecificB1, networkSpecificB2}|
           resetTimer  {networkSpecificB1, networkSpecificB2}|
           statusReport  {networkSpecificB1, networkSpecificB2}|
           scriptClose  {networkSpecificB1, networkSpecificB2}|
           scriptEvent  {networkSpecificB1, networkSpecificB2}|
           scriptInformation  {networkSpecificB1, networkSpecificB2}|
           scriptRun  {networkSpecificB1, networkSpecificB2}|
           sendChargingInformation  {networkSpecificB1, networkSpecificB2}|
           specializedResourceReport |
           promptAndReceiveMessage {networkSpecificB1, networkSpecificB2, networkSpecificB3 }
           }
```

```
AssistHandoffSsfToScfReturnable OPERATION::= {
            activityTest |
            applyCharging {networkSpecificB1, networkSpecificB2}|
            applyChargingReport {networkSpecificB1, networkSpecificB2}|
            assistRequestInstructions {networkSpecificB1, networkSpecificB2}|
            cancel {networkSpecificB2}|
            cancelStatusReportRequest {networkSpecificB1, networkSpecificB2}|
            connectToResource {networkSpecificB1, networkSpecificB2}|
            disconnectForwardConnection |
            disconnectForwardConnectionWithArgument {networkSpecificB1, networkSpecificB2}|
            furnishChargingInformation {networkSpecificB1, networkSpecificB2}|
            playAnnouncement {networkSpecificB1, networkSpecificB2, networkSpecificB3 }|
            promptAndCollectUserInformation {networkSpecificB1, networkSpecificB2,
            networkSpecificB3 }|
            requestCurrentStatusReport {networkSpecificB1, networkSpecificB2}|
            requestEveryStatusChangeReport {networkSpecificB1, networkSpecificB2}|
            requestFirstStatusMatchReport {networkSpecificB1, networkSpecificB2}|
            resetTimer {networkSpecificB1, networkSpecificB2}|
            scriptClose {networkSpecificB1, networkSpecificB2}|
            scriptInformation {networkSpecificB1, networkSpecificB2}|
            scriptRun {networkSpecificB1, networkSpecificB2}|
            promptAndReceiveMessage {networkSpecificB1, networkSpecificB2, networkSpecificB3 }
            }


scf-ssfGenericAbstractSyntax  ABSTRACT-SYNTAX::= {
    GenericSCF-SSF-PDUs
    IDENTIFIED BY   id-as-scf-ssfGenericAS}

GenericSCF-SSF-PDUs::= TCMessage {{ScfToSsfGenericInvokable}, {ScfToSsfGenericReturnable}}

ScfToSsfGenericInvokable  OPERATION::= {
            activateServiceFiltering {networkSpecificB1, networkSpecificB2, networkSpecificB3}|
            activityTest |
            applyCharging {networkSpecificB1, networkSpecificB2}|
            applyChargingReport {networkSpecificB1, networkSpecificB2}|
            callInformationRequest {networkSpecificB1}|
            cancel {networkSpecificB2}|
            cancelStatusReportRequest {networkSpecificB1, networkSpecificB2}|
            collectInformation {networkSpecificB1 }|
            connect {networkSpecificB1, networkSpecificB2}|
            connectToResource {networkSpecificB1, networkSpecificB2}|
            continue |
            continueWithArgument{networkSpecificB1, networkSpecificB2}|
            createCallSegmentAssociation {networkSpecificB1, networkSpecificB2}|
            disconnectForwardConnection |
            disconnectForwardConnectionWithArgument {networkSpecificB1, networkSpecificB2}|
            disconnectLeg {networkSpecificB1, networkSpecificB2}|
            establishTemporaryConnection {networkSpecificB1, networkSpecificB2}|
            furnishChargingInformation  {networkSpecificB1, networkSpecificB2}|
            initiateCallAttempt  {networkSpecificB1, networkSpecificB2}|
            mergeCallSegments {networkSpecificB1, networkSpecificB2}|
            moveCallSegments {networkSpecificB1, networkSpecificB2}|
            moveLeg {networkSpecificB1, networkSpecificB2}|
            releaseCall { networkSpecificB2}|
            requestCurrentStatusReport  {networkSpecificB1, networkSpecificB2}|
            requestEveryStatusChangeReport {networkSpecificB1, networkSpecificB2}|
            requestFirstStatusMatchReport  {networkSpecificB1, networkSpecificB2}|
            requestNotificationChargingEvent  {networkSpecificB2}|
            requestReportBCSMEvent  {networkSpecificB1, networkSpecificB2}|
            requestReportUTSI {networkSpecificB1, networkSpecificB2}|
            resetTimer  {networkSpecificB1, networkSpecificB2}|
            sendChargingInformation  {networkSpecificB1, networkSpecificB2}|
            sendSTUI {networkSpecificB1, networkSpecificB2}|
            setServiceProfile {networkSpecificB1, networkSpecificB2}|
            splitLeg  {networkSpecificB1, networkSpecificB2}|
            playAnnouncement  {networkSpecificB1, networkSpecificB2, networkSpecificB3 }|
            promptAndCollectUserInformation  {networkSpecificB1, networkSpecificB2,
            networkSpecificB3 }|
            scriptClose  {networkSpecificB1, networkSpecificB2}|
            scriptInformation  {networkSpecificB1, networkSpecificB2}|
            scriptRun  {networkSpecificB1, networkSpecificB2}|
            promptAndReceiveMessage {networkSpecificB1, networkSpecificB2, networkSpecificB3 }
            }
```

```
ScfToSsfGenericReturnable  OPERATION::= {
            activateServiceFiltering  {networkSpecificB1, networkSpecificB2, networkSpecificB3}|
            activityTest |
            applyCharging  {networkSpecificB1, networkSpecificB2}|
            applyChargingReport  {networkSpecificB1, networkSpecificB2}|
            callInformationReport  {networkSpecificB1, networkSpecificB2}|
            callInformationRequest  {networkSpecificB1}|
            cancel  {networkSpecificB2}|
            cancelStatusReportRequest  {networkSpecificB1, networkSpecificB2}|
            collectInformation  {networkSpecificB1 }|
            connect  {networkSpecificB1, networkSpecificB2}|
            connectToResource  {networkSpecificB1, networkSpecificB2}|
            createCallSegmentAssociation {networkSpecificB1, networkSpecificB2}|
            disconnectForwardConnection |
            disconnectForwardConnectionWithArgument {networkSpecificB1, networkSpecificB2}|
            disconnectLeg {networkSpecificB1, networkSpecificB2}|
            entityReleased {networkSpecificB2}|
            establishTemporaryConnection {networkSpecificB1, networkSpecificB2}|
            eventNotificationCharging {networkSpecificB1, networkSpecificB2} |
            resetTimer {networkSpecificB1, networkSpecificB2}|
            eventReportBCSM  {networkSpecificB1, networkSpecificB2}|
            furnishChargingInformation  {networkSpecificB1, networkSpecificB2}|
            initiateCallAttempt  {networkSpecificB1, networkSpecificB2}|
            mergeCallSegments {networkSpecificB1, networkSpecificB2}|
            moveCallSegments {networkSpecificB1, networkSpecificB2}|
            moveLeg {networkSpecificB1, networkSpecificB2}|
            reportUTSI {networkSpecificB1, networkSpecificB2}|
            requestCurrentStatusReport  {networkSpecificB1, networkSpecificB2}|
            requestEveryStatusChangeReport {networkSpecificB1, networkSpecificB2}|
            requestFirstStatusMatchReport  {networkSpecificB1, networkSpecificB2}|
            requestNotificationChargingEvent  {networkSpecificB2}|
            requestReportBCSMEvent  {networkSpecificB1, networkSpecificB2}|
            requestReportUTSI {networkSpecificB1, networkSpecificB2} |
            sendChargingInformation  {networkSpecificB1, networkSpecificB2}|
            sendSTUI {networkSpecificB1, networkSpecificB2}|
            serviceFilteringResponse  {networkSpecificB1, networkSpecificB2}|
            setServiceProfile {networkSpecificB1, networkSpecificB2}|
            splitLeg {networkSpecificB1, networkSpecificB2}|
            statusReport  {networkSpecificB1, networkSpecificB2}|
            playAnnouncement  {networkSpecificB1, networkSpecificB2, networkSpecificB3 }|
            promptAndCollectUserInformation  {networkSpecificB1, networkSpecificB2,
            networkSpecificB3 }|
            scriptClose  {networkSpecificB1, networkSpecificB2}|
            scriptEvent  {networkSpecificB1, networkSpecificB2}|
            scriptInformation  {networkSpecificB1, networkSpecificB2}|
            scriptRun  {networkSpecificB1, networkSpecificB2}|
            specializedResourceReport |
            promptAndReceiveMessage  {networkSpecificB1, networkSpecificB2, networkSpecificB3 }
            }




scf-ssfTrafficManagementAbstractSyntax  ABSTRACT-SYNTAX::= {
    TrafficManagementSCF-SSF-PDUs
    IDENTIFIED BY   id-as-scf-ssfTrafficManagementAS}

TrafficManagementSCF-SSF-PDUs::= TCMessage {{ScfToSsfTrafficManagementInvokable}, {EmptyReturnable}}

ScfToSsfTrafficManagementInvokable OPERATION::= {
    callGap {networkSpecificB1, networkSpecificB2, networkSpecificB3}
    }


scf-ssfServiceManagementAbstractSyntax  ABSTRACT-SYNTAX::= {
    ServiceManagementSCF-SSF-PDUs
    IDENTIFIED BY   id-as-scf-ssfServiceManagementAS}

ServiceManagementSCF-SSF-PDUs::= TCMessage {{ScfToSsfServiceManagementInvokable},
            {ScfToSsfServiceManagementReturnable}}

ScfToSsfServiceManagementInvokable OPERATION::= {
    activateServiceFiltering {networkSpecificB1, networkSpecificB2, networkSpecificB3}
    }

ScfToSsfServiceManagementReturnable OPERATION::= {
    activateServiceFiltering {networkSpecificB1, networkSpecificB2, networkSpecificB3}
    }
```

```
ssf-scfServiceManagementAbstractSyntax  ABSTRACT-SYNTAX::= {
    ServiceManagementSSF-SCF-PDUs
    IDENTIFIED BY   id-as-ssf-scfServiceManagementAS}


ServiceManagementSSF-SCF-PDUs::= TCMessage {{SsfToScfServiceManagementInvokable}, {EmptyReturnable}
}


SsfToScfServiceManagementInvokable OPERATION::= {
    serviceFilteringResponse {networkSpecificB1, networkSpecificB2}
    }



scf-ssfStatusReportingAbstractSyntax  ABSTRACT-SYNTAX::= {
    StatusReportingSCF-SSF-PDUs
    IDENTIFIED BY   id-as-scf-ssfStatusReportingAS}


StatusReportingSCF-SSF-PDUs::= TCMessage {{ScfToSsfStatusReportingInvokable},
            {ScfToSsfStatusReportingReturnable}}


ScfToSsfStatusReportingInvokable OPERATION ::= {

            cancelStatusReportRequest  {networkSpecificB1, networkSpecificB2}|
            requestCurrentStatusReport  {networkSpecificB1, networkSpecificB2}|
            requestEveryStatusChangeReport {networkSpecificB1, networkSpecificB2}|
            requestFirstStatusMatchReport {networkSpecificB1, networkSpecificB2}
            }


ScfToSsfStatusReportingReturnable OPERATION ::= {

            cancelStatusReportRequest  {networkSpecificB1, networkSpecificB2}|
            requestCurrentStatusReport  {networkSpecificB1, networkSpecificB2}|
            requestEveryStatusChangeReport {networkSpecificB1, networkSpecificB2}|
            requestFirstStatusMatchReport  {networkSpecificB1, networkSpecificB2}|
            statusReport {networkSpecificB1, networkSpecificB2}
            }


scf-ssfTriggerManagementAbstractSyntax  ABSTRACT-SYNTAX::= {
    TriggerManagementSCF-SSF-PDUs
    IDENTIFIED BY   id-as-scf-ssfTriggerManagementAS}


TriggerManagementSCF-SSF-PDUs::= TCMessage {{ScfToSsfTriggerManagementInvokable},
            {ScfToSsfTriggerManagementReturnable}}


ScfToSsfTriggerManagementInvokable OPERATION::= {
    createOrRemoveTriggerData {networkSpecificB1, networkSpecificB2, networkSpecificB3}|
    manageTriggerData {networkSpecificB1, networkSpecificB2}
    }


ScfToSsfTriggerManagementReturnable OPERATION::= {
    createOrRemoveTriggerData {networkSpecificB1, networkSpecificB2, networkSpecificB3}|
    manageTriggerData {networkSpecificB1, networkSpecificB2}
    }

END
```

15 Services assumed from TCAP

15.1 Introduction

The common procedures and mapping which apply between INAP and TC to be used are defined in EN 301 931-1. This part describes the specific procedures and mapping instructions for the INAP SSF – SCF interfaces as defined in subsequent clauses.

15.1.1 SSF-SCF Interface

15.1.1.1 Normal Procedures

15.1.1.1.1 SSF-to-SCF messages

This clause defines the normal procedures for TC messages from the SSF to the SCF.

15.1.1.1.1.1 SSF-FSM related messages

A dialogue shall be established when the SSF-FSM moves from the state **Idle** to the state **Waiting for Instructions**. The relevant INAP operation, which can be the InitialDP operation for TDP-R, shall be transmitted in the same message.

For all other operations sent from the SSF-FSM, the dialogue shall be maintained except for the following cases.

When the SSF-FSM makes a non-error case state transition to the state **Idle** and there is one or more pending operation and TCAP dialogue is established, TCAP dialogue can be terminated by TC-END primitive with component(s). When the SSF sends the last EventReportBCSM, ApplyChargingReport or CallInformationReport the dialogue may be ended from the SSF by a TC-END request primitive with basic end.

In the case that there is no pending operation and TCAP dialogue is established, TCAP dialogue can be terminated by TC-END primitive with zero component or prearranged end. When the SSF-FSM makes a non-error case state transition to the state **Idle** and there is no operation to be sent, the dialogue is ended by means of a TC-END request primitive (basic) with zero components, or the dialogue is locally ended by means of a TC-END request primitive with prearranged end.

The SSF can end a dialogue with a TC-END request primitive with zero component or prearranged end depending on that TCAP dialogue is established or not, in the case call release is initiated by any other entity than the SCF and the SSF has no pending call information requests (or pending requests which should be treated in the same way) nor any armed EDP to notify the SCF of the call release (for alternative way, see "Abnormal Procedures" 15.1.1.2).

When the SSF has sent the last EventReportBCSM, ApplyChargingReport or CallInformationReport the dialogue may be ended from the SCF by a TC-END request primitive with basic end.

15.1.1.1.1.2 Assisting/Hand-off SSF FSM related messages

A dialogue shall be established when the Assisting/Hand-off SSF-FSM moves from the state **Idle** to the state **Waiting for Instructions**. The AssistRequestInstructions operation shall be transmitted with a TC-BEGIN request primitive.

For all other operations sent from the Assisting/Hand-off SSF-FSM, the dialogue shall be maintained except for the following cases.

When the SSF-FSM makes a non-error case state transition to the state **Idle** and there is one or more pending operation and TCAP dialogue is established, TCAP dialogue can be terminated by TC-END primitive with component(s). When the SSF sends the last ApplyChargingReport, the dialogue may be ended from the SSF by a TC-END request primitive with basic end.

In the case that there is no pending operation and TCAP dialogue is established, TCAP dialogue can be terminated by TC-END primitive with zero component or prearranged end. When the SSF-FSM makes a non-error case state transition to the state **Idle** and there is no operation to be sent, the dialogue is ended by means of a TC-END request primitive (basic) with zero components, or the dialogue is locally ended by means of a TC-END request primitive with prearranged end.

When the SSF has sent the last ApplyChargingReport, the dialogue may be ended from the SCF by a TC-END request primitive with basic end.

15.1.1.1.1.3   SSME-FSM related messages

The following procedures shall be followed:

- The dialogue shall be maintained when the ActivityTest Return Result is sent.

- No dialogue shall be established when the ServiceFilteringResponse operation is sent. The operation is sent with a TC-BEGIN request primitive and the dialogue is ended by means of a TC-END request primitive with prearranged end.

- A dialogue shall no longer be maintained when the Return Result of the ActivateServiceFiltering operation is sent. The dialogue is ended by means of a TC-END request primitive with basic end, the Return Result is transmitted with the same request.

- The dialogue is locally terminated by means of a TC-END request primitive with prearranged end, upon reception of a TC-BEGIN indication primitive with a CallGap operation.

- The dialogue shall be maintained when the RequestCurrentStatusReport, RequestEveryStatusChangeReport, RequestFirstStatusMatchReport, or CancelStatusReportRequest operation is received inside the call context.

- The dialogue shall be maintained when the following response or operation is sent:

  - Return Result of the RequestEveryStatusChangeReport;

  - Return Result of the RequestFirstStatusMatchReport; and

  - StatusReport operation for reporting the resource status change in reply to the RequestEveryStatusChangeReport.

- The dialogue shall be maintained on sending the following response or operation inside the call context if the response or operation is not the final one:

  - Return Result of the RequestCurrentStatusReport operation; and

  - StatusReport operation in reply to the RequestFirstStatusMatchReport operation.

- The dialogue shall no longer be maintained on sending the following response or operation inside the call context if the response or operation is the final one:

  - Return Result of the RequestCurrentStatusReport operation; and

  - StatusReport operation in reply to the RequestFirstStatusMatchReport operation.

The dialogue is ended from the SSF by means of a TC-END request primitive with basic end, one of above response or operation is transmitted with the same request.

- If the monitor duration expires or the monitor is cancelled for RequestFirstStatusMatchReport or RequestEveryStatusChangeReport which has been received inside the call context and there is no needs to maintain dialogue, the dialogue is ended from the SSF by means of a TC-END request primitive (basic) with a StatusReport operation.

- The dialogue shall be established when the RequestCurrentStatusReport, RequestEveryStatusChangeReport or RequestFirstStatusMatchReport operation is received outside the call context.

- The dialogue shall be maintained when the CancelStatusReportRequest operation is received outside the call context.

- The dialogue shall no longer be maintained when the Return Result of the RequestCurrentStatusReport operation is sent outside the call context. The dialogue is ended from the SSF by means of a TC-END request primitive with basic end, the Return Result is transmitted with the same request.

- The dialogue shall be maintained when the Return Result of the RequestFirstStatusMatchReport or RequestEveryStatusChangeReport operation is sent outside the call context.

- The dialogue shall no longer be maintained when the StatusReport operation for reporting the resource status match is sent in reply to the RequestFirstStatusMatchReport operation outside the call context.
  The dialogue is ended from the SSF by means of a TC-END request primitive with basic end, the StatusReport operation is transmitted with the same request.

- The dialogue shall be maintained when the StatusReport operation for the resource status change is sent in reply to the RequestEveryStatusChangeReport operation outside the call context.

- The dialogue shall no longer be maintained when the monitor duration expires or the monitor is cancelled for RequestFirstStatusMatchReport or RequestEveryStatusChangeReport which has been received outside the call context. The dialogue is ended from the SSF by means of a TC-END request primitive (basic) with a StatusReport operation.

15.1.1.1.2 SCF-to-SSF messages

This clause defines the normal procedures for TC messages from the SCF to the SSF.

15.1.1.1.2.1 SCSM-FSM related messages

A dialogue shall be established when the SCSM-FSM moves from state Idle to state Preparing SSF Instructions upon the receipt of InitialDP operation for TDP-R, or AssistRequestInstructions operation.

A dialogue shall be established when the SCSM-FSM sends an InitiateCallAttempt or a CreateCSA from the **Idle** state.

For subsequent operations sent from the SCSM-FSM, the dialogue shall be maintained except for the following cases, i.e. all other operations are sent after a dialogue was established from the SSF (the SCF has previously received a TC-BEGIN indication primitive with an InitialDP operation or an AssistRequestInstructions operation).

The dialogue shall no longer be maintained when the prearranged end condition is met in the SCF. When the SCF does not expect any messages other than possibly REJECT or ERROR messages for the operations sent and when the last associated operation timer expires, the dialogue is locally ended by means of a TC-END request primitive with prearranged end.

Alternatively, the sending of operations, leading to the termination of the relationship, by means of a TC-END request primitive (basic end) is possible.

15.1.1.1.2.2 SCME related messages

The following procedures shall be followed:

- The dialogue shall be maintained when the ActivityTest operation is sent.

- The dialogue shall be maintained when the ActivityTest result is sent.

- A dialogue shall not be established when a CallGap operation is sent without using a SCSM associated dialogue. The operation is sent using a TC-BEGIN request primitive and the dialogue is terminated with a prearranged end.

- For sending one or more CallGap operations, the SCME may use an existing SCSM FSM associated dialogue which was initiated by a SSF-FSM (i.e. established for the transmission of the InitialDP operation). The dialogue shall be maintained and the CallGap operation(s) shall be sent with the first response of the SCSM FSM to the InitialDP operation.

- A dialogue shall be established when an ActivateServiceFiltering operation is sent. The operation shall be transmitted with a TC-BEGIN request primitive.

- The dialogue is locally terminated upon reception of a ServiceFilteringResponse operation using a TC-END request primitive with prearranged end.

- The dialogue shall be maintained when the RequestCurrentStatusReport, RequestEveryStatusChangeReport, RequestFirstStatusMatchReport, or CancelStatusReportRequest operation is sent inside the call context.

- The dialogue shall be maintained when the following response or operation is received:

    - ReturnResult of the RequestEveryStatusChangeReport;

    - ReturnResult of the RequestFirstStatusMatchReport; and

    - StatusReport operation for reporting the resource status change in reply to the RequestEveryStatusChangeReport.

- The dialogue shall be maintained on receiving the following response or operation inside the call context if the response or operation is not final one:

    - Return Result of the RequestCurrentStatusReport operation; and

    - StatusReport operation in reply to the RequestFirstStatusMatchReport operation.

- The dialogue shall no longer be maintained on receiving the following response or operation inside the call context if the response or operation is final one:

    - Return Result of the RequestCurrentStatusReport operation; and

    - StatusReport operation in reply to RequestFirstStatusMatchReport operation.

The dialogue is ended from the SSF by means of a TC-END indication primitive with basic end, one of above response or operation is transmitted with the same indication.

- If the monitor duration expires or the monitor is cancelled for RequestFirstStatusMatchReport or RequestEveryStatusChangeReport which has been sent inside the call context and there is no needs to maintain dialogue, the dialogue is ended from the SSF by means of a TC-END indication primitive (basic) with a StatusReport operation.

- The dialogue shall be established when the RequestCurrentStatusReport, RequestEveryStatusChangeReport or RequestFirstStatusMatchReport operation is sent outside the call context.

- The dialogue shall be maintained when the CancelStatusReportRequest operation is sent outside the call context.

- The dialogue shall no longer be maintained when the Return Result of the RequestCurrentStatusReport operation is received outside the call context. The dialogue is ended from the SSF by means of a TC-END indication primitive with basic end, the Return Result is transmitted with the same indication.

- The dialogue shall be maintained when the Return Result of the RequestFirstStatusMatchReport or RequestEveryStatusChangeReport operation is received outside the call context.

- The dialogue shall no longer be maintained when the StatusReport operation for reporting the resource status match is received in reply to the RequestFirstStatusMatchReport operation outside the call context. The dialogue is ended from the SSF by means of a TC-END indication primitive with basic end, the StatusReport operation is received with the same indication.

- The dialogue shall be maintained when the StatusReport operation for reporting the resource status change is received in reply to the RequestEveryStatusChangeReport operation outside the call context.

- The dialogue shall no longer be maintained when the monitor duration expires or the monitor is cancelled for RequestFirstStatusMatchReport or RequestEveryStatusChangeReport outside the call context. The dialogue is ended from the SSF by means of a TC-END indication primitive (basic) with a StatusReport operation.

- The dialogue shall be established when the ManageTriggerData operation is sent. This operation is sent outside the call context.

- The dialogue shall no longer be maintained when the ManageTriggerDataResult of the ManageTriggerData operation is received. The dialogue is ended from the SSF by means of a TC-END indication primitive with basic end, the ManageTriggerDataResult is transmitted with the same indication.

- The dialogue shall be established when the CreateOrRemoveTriggerData operation is sent. This operation is sent outside the call context.

- The dialogue shall no longer be maintained when the CreateOrRemoveTriggerDataResult of the CreateOrRemovetriggerData operation is received. The dialogue is ended from the SSF by means of a TC-END indication primitive with basic end, the CreateOrRemoveTriggerDataResult is transmitted with the same indication.

15.1.1.1.2.3 SCF-SSF - Use of dialogue handling services

Dialogue handling services are used to trigger the sending of the APDUs associated with the operations involved in the INAP packages.

Component grouping is performed under the control of the application-process through an appropriate usage of the TC-BEGIN and TC-CONTINUE service.

The TC-END service is solely used to support the dialogue closing procedure (i.e. it is never used to trigger the sending of components).

On receipt of an empty TC-CONTINUE.req primitive, the FE should ignore the primitive.

On receipt of an TC-END.req with a INAP request, the FE should not perform the request and consider the requested TC-END service as a dialogue closing procedure. The dialogue is then terminated.

It is an application-process responsibility to provide in the TC-BEGIN.req primitive a destination address which can be used by the underlying SCCP to route the message to the proper FE if this FE is addressed through the SS7 network.

The prearranged end can be used.

15.1.1.2 Abnormal Procedures

15.1.1.2.1 SCF-to-SSF messages

Considering that SSF do not have the logic to recover from error cases detected on the SCF-SSF interface, the following shall apply:

- Operation errors and rejection of TCAP components shall be transmitted to the SSF with a TC-END request primitive, basic end.

- If, in violation of the above procedure, an ERROR or REJECT component is received with a TC-CONTINUE indication primitive, the SSF shall abort the dialogue with a TC-U-ABORT request primitive.

- In the case of the SSF relay, it is outside the scope of this capability set how to map messages to ROSE capability of bearer signalling system between the SSF and the SRF, and what services are assumed from ROSE.

15.1.1.2.2 SSF-to-SCF messages

Operation errors and rejection of TCAP components shall be transmitted to the SCF according to the following rules:

- The dialogue shall be maintained when the preceding message, which contained the erroneous component, indicated that the dialogue shall be maintained, i.e. the error or reject shall be transmitted with a TC-CONTINUE request primitive if the erroneous component was received with a TC-CONTINUE indication primitive.

- On receipt of an ERROR or REJECT component the SCF decides on further processing. It may either continue, explicitly end or abort the dialogue.

- In all other situations the dialogue shall no longer be maintained, i.e. the error or reject shall be transmitted with a TC-END request primitive, basic end, if the erroneous component was received with a TC-BEGIN indication primitive.

- On expiration of application timer T_{SSF}, dialogue shall be terminated by means of by TC-U-ABORT primitive with an Abort reason, regardless of TCAP dialogue is established or not.

If the error processing in the SSF leads to the case where the SSF is not able to process further SCF operations while the dialogue is to be maintained, the SSF aborts the dialogue with a TC-END request primitive with basic end or a TC-U-ABORT request primitive, depending on whether any pending ERROR or REJECT component is to be sent or not.

The SSF can end a dialogue with a TC-U-ABORT request primitive in case call release is initiated by any other entity then the SCF and the SSF has no pending call information requests (or pending requests which should be treated in the same way, i.e. ApplyCharging nor any armed EDP to notify the SCF of the call release (for alternative way, see "Normal Procedures"15.1.1.1).

In the case of the SSF relay, it is outside the scope of this capability set how to map messages to ROSE capability of bearer signalling system between the SSF and the SRF, and what services are assumed from ROSE.

15.1.1.2.3 SCF-SSF - Use of dialogue handling services

On receipt of a TC-U-REJECT.ind in the FE, this primitive should be ignored. It is up to the application process to abort, continue or terminate the dialogue, if not already terminated by the sending application process according to the rules as stated herein. This is also applicable for invoke problems related to a class 4 linked operation.

A TC-U-REJECT.req should be sent followed by a TC-CONTINUE.req.

On receipt of a TC-R-REJECT.ind in the FE, this primitive should be ignored. It is up to the application process to abort, continue or terminate the dialogue, if not already terminated by the sending application process according to the rules as stated herein. This is also applicable for invoke problems related to a class 4 linked operation. The dialogue should be released with a TC-U-ABORT.req.

On receipt of a TC-L-REJECT indication primitive (i.e. when a protocol error has been detected by the local TC entity) which cannot be related to an active operation, it is up to the application process to continue or to terminate the dialogue and implicitly trigger the transmission of the reject component or to abort the dialogue.

On receipt of a TC-NOTICE indication the SACF is informed that a message cannot be delivered by the Network Layer. It occurs if the Return Option has been set. It is for the application process to decide whether to terminate the dialogue or retry.

The application-process is the sole user of the TC-P-ABORT service and TC-NOTICE service.

The receipt of a TC-U-ABORT-Ind or TC-P-ABORT-Ind on a dialogue terminates all request processing.

15.1.1.3 Dialogue Handling

Refer to common procedures defined in EN 301 931-1.

15.1.1.3.1 Dialogue Establishment

Refer to common procedures defined in EN 301 931-1.

15.1.1.3.2 Dialogue Continuation

Refer to common procedures defined in EN 301 931-1.

15.1.1.3.3 Dialogue Termination

Refer to common procedures defined in EN 301 931-1.

15.1.1.3.4 User Abort

Refer to common procedures defined in EN 301 931-1.

15.1.1.3.5   Provider Abort

Refer to common procedures defined in EN 301 931-1.

15.1.1.3.6   Mapping to TC Dialogue Primitives

The SSF-SCF IN services can be mapped onto TC services. This clause defines the mapping of the SSF-SCF IN services onto the services of the TC dialogue handling services defined in ETS 300 287-1.

a) The TC-BEGIN service is used to invoke the operations of the **scf-ssfConnectionPackage** and **ssf-scfConnectionPackage**.

b) The TC-CONTINUE service is used to report the success of the operations invoked in a TC-BEGIN service and to invoke or respond to any other operations.

c) The TC-U-ABORT service is used to report the failure of operations of the **scf-ssfConnectionPackage** and **ssf-scfConnectionPackage**.

The mapping of the parameters onto the TC-BEGIN primitive is defined in EN 301 931-1 "Mapping to TC dialogue primitives" with the following qualifications:

The Application Context Name parameter shall take the value of the application-context-name field of the **cs2ssf-scfGenericAC**, **cs2ssf-scfAssistHandoffAC** or **cs2ssf-scfServiceManagementAC** object if the initiating AE is a SSF or the **cs2scf-ssfGenericAC,**, **cs2scf-ssfTrafficManagementAC**, **cs2scf-ssfServiceManagementAC** or **cs2scf-ssfStatusReportingAC** object if the originating AE is a SCF.

The mapping of the parameters onto the TC-CONTINUE primitive is defined in EN 301 931-1 "Mapping to TC dialogue primitives".

The mapping of the parameters onto the TC-U-ABORT primitive is defined in EN 301 931-1 "Mapping to TC dialogue primitives" with the following qualifications:

The Application-Context-Name parameter shall be used as specified in ETS 300 287-1. When the responding AE refuses a dialogue because the application-context-name it receives is not supported, this parameter shall have the value of the application-context-name field of the **cs2ssf-scfGenericAC**, **cs2ssf-scfAssistHandoffAC,** **cs2ssf-scfServiceManagementAC** or **cs2scf-ssfGenericAC** object if the responding AE is a SCF or the, **cs2scf-ssfTrafficManagementAC**, **cs2scf-ssfServiceManagementAC** or **cs2scf-ssfStatusReportingAC** object if the responding AE is a SSF.

The use of the parameters of the TC-END service is defined in EN 301 931-1 "Mapping to TC dialogue primitives".

15.1.1.4   Component Handling

15.1.1.4.1   Procedures for INAP Operations

The INAP ASEs are users of the TC component handling services except for the TC-L-REJECT and TC-L-CANCEL services which are used by the application-process. Receipt of a TC-L-REJECT-Ind leads the application-process to abandon the dialogue (i.e. it issues a TC-U-ABORT-Request primitive).

The TC-U-CANCEL service is never used.

15.1.1.4.2   Mapping to TC Component Parameters

The SSF-SCF IN ASE services are mapped onto the TC component handling services. The mapping of operations and errors onto TC services is defined in EN 301 931-1 "Mapping to TC components primitives" with the following qualifications:

The timeout parameter of the TC-INVOKE-Req primitives is set according to the requirements set out in the CCF/SSF -SCF interface clause.

16   Charging Scenarios supported by Core INAP

16.1   Introduction

Information is provided on how the different charging capabilities in INAP CS3 may be used. Networks may support different or additional charging capabilities then listed herein.

With the introduction of IN, the charging as performed by the basic call process has to be extended. With IN, charging processes can be activated in both SCPs and SSPs. When for an IN call the charging processes in the SCP have to interwork with the charging processes in the SSP and the basic network (e.g. PSTN), specific charging operations have to be transferred via the INAP. The concept of charging from an IN point of view is described. First some terminology concerning charging processes and charging capabilities is listed, followed by particular charging scenarios for which the use of the INAP operations is explained.

16.2   Terminology

Because there is already an existing definition of the terms Charge Determination and Charge Generation within the ES 201 296 which does not cover all aspects of IN Charge Determination and Generation the terms Charge Data Determination, Charge Data Generation and Charge Data Registration are used.

Charge Data Determination (DET)

All activities to determine charging or billing for an IN call. Following determinations may be distinguished:

- party(s) to be charged for the network access and/or IN service usage. Charge party can be the calling line or IN service subscriber or both;

- tariff- respectively level of charge;

  NOTE: Through the whole ETSI CS1 the term 'charge level' is used and never explained. It is assumed, that the term is a general term covering both: the 'charge band' ('pointer to tariff') as it is usual defined at national ISUP and a real tariff (as defined within ES 201 296).

- determining the items to be charged;

- method of charging the party(s) to be charged: type of charging/billing records, off-line or on-line.

Note that different nodes (functions) may be involved in charge determination.

If determination of the previous elements is performed off-line then in that case only a call record is registered containing call- and service related data.

Charge Data Generation (GEN)

Time dependent generation of the correct charges. In case of off-line charging the data collection (e.g. time stamps, call duration).

Charge Data Registration (REG)

Registering the evaluated charges (on-line) and/or the collected charge data (off-line) into call records for providing them to a postprocessing centre or updating the charge meters or both.

Charge Output (OUT)

Output of charging data for further processing. Charging data can be output to magnetic tapes or data-links, on operator request or scheduled.

Off-line charging

The usage and/or charge information of the call is recorded in the network (e.g. OLE, SSP or SCP). The calculation of the charge for that call and the billing is performed in an off-line process. The information recorded could also be used for other purposes by the network operator (e.g. accounting).

Off-line charging/billing/accounting process (OFC)

A FE which processes the call records retrieved from the other FEs (SSF, SCF, international exchange, LE) to prepare the bill for the subscriber or to support other accounting processes.

On-line charging

In this case charging information during the call instance has to be calculated in real time. This further processing of charging information in real time could be for the support of the pay phone, AOC (advice of charge) or for charge metering or for credit limit supervision.

On-line Charge Information provision to the user access (ONC)

Provision of charge pulses or signalling information on the user/network interface during call instance (e.g. via ISDN: Advice of Charge (AOC)).

In the PSTN/PLMN it is in general regarded as subscriber feature.

16.3   Charging scenarios

In an IN structured network, the charging for services may be split between several parties. Each of the following scenarios shows a possible atomic charging configuration for one of the parties. Scenarios may be combined to give the total charging capabilities required for a service. The choice of scenario for each charged party is a network specific option.

Dependent on the location of Charge Data Determination, Charge Data Generation and Charge Data Registration different IN charging scenarios may be distinguished. In the following these atomic scenarios are listed.

For simplification the model 'SSP on transit level with incoming- and outgoing ISUP signalling and charging of the calling line is done at the originating local exchange (OLE)' is used. Of course the same principles apply if the SSP is located at an other level within the basic network. For outgoing ISUP signalling an succeeding exchange (SE) is used. Function related to this SE may be located in any type of succeeding exchange, e.g. transit exchange.

There is no special distinction between on- and off-line charging made in the following because the scenarios may almost be the same for both methods.

16.3.1   Scenario A: Application of the Basic Network Charging Function (BNCF)

For call related IN services a basic network (PSTN/PLMN) may be used for accessing the IN. The charging for that access is usually done by the standard network charging functions (e.g. 'calling line').

In one case (**scenario A.1**), the charging is completely done by the existing charging mechanisms in the PSTN/ISDN, such as using the service access code to determine the tariff, and meters in the LE to count the charge pulses. For this mechanism, no INAP operations are required, as no charging functions are performed by the SSF, SCF or any other IN FE.
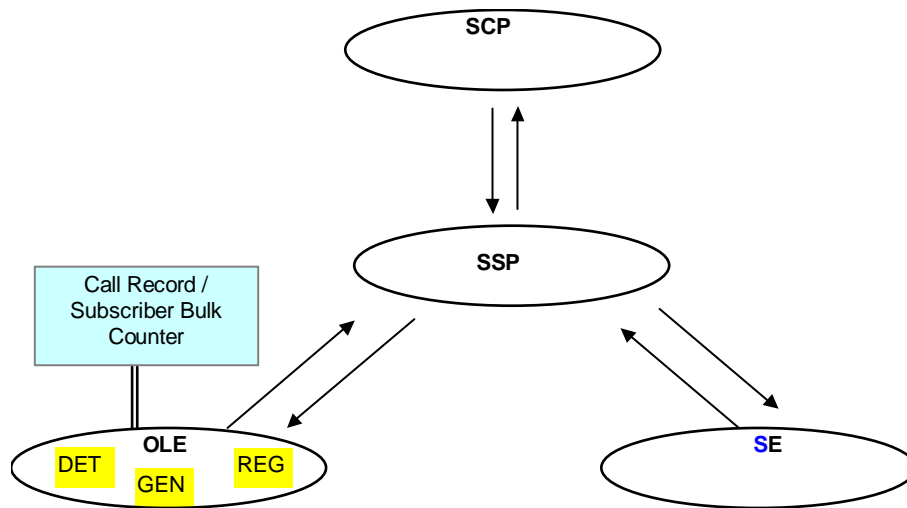
Figure 63: Scenario A.1

In the other case, the SCF has control of the charging information and instructs the SSF on the charging information to be sent by the SSF (**scenario A.2**).

In the LE, either a charge meter can be updated or a standard call record can be generated. There is no call record generated at the SSF or SCF.
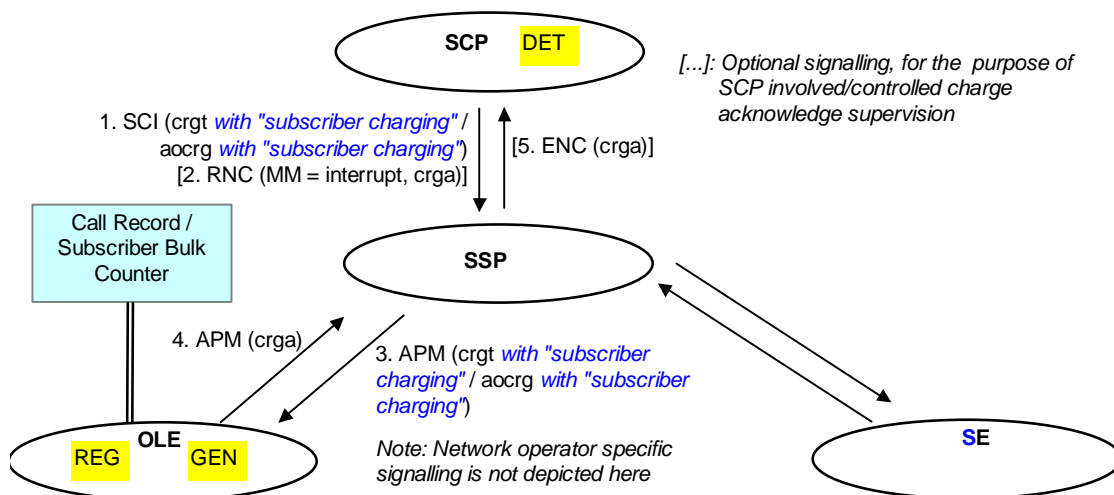

Figure 64: Scenario A.2

If the SSF is a LE, the principles are the same, but the SSF-LE interface will be internal rather than by network signalling. The SCF needs not to know whether the SSP is a transit or a local exchange.

16.3.2   Scenario B: IN charging completely in the IN

In this scenario, the charging is done completely in the IN nodes (SSF and/or SCF). The PSTN will determine from e.g. the service access code, that no charge is to be raised, and all accounting will be performed at either the SSF or the SCF. The control of charging is always at the SCF, but call records may be registered at either the SSF or SCF.

In case call records are registered at the SSF charge data generation may be done completely at the SSF or at both the SSF and the SCF (**scenario B.1**). Data collection may be done at both components, e.g. collection of usual call related charge data at the SSF and collection of data which result on an user interactive dialogue or application of special service features at the SCF.
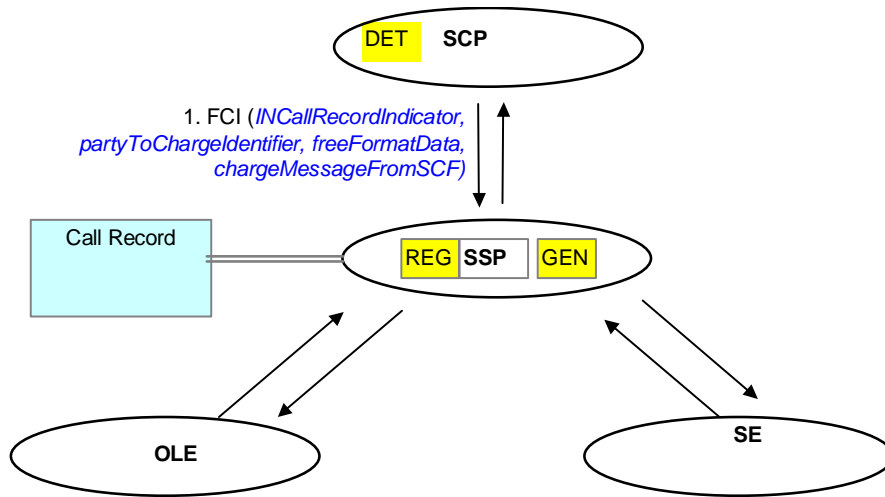
Figure 65: Scenario B.1

The same cases can be distinguished for the SCF registration. In this case charge data generation may be done at the SSF (**scenario B.2**) or completely at the SCF (referred to as **scenario B.3 - not shown**). For scenario B.2 the SCF may add additional charges e.g. for service feature usage, special announcements.
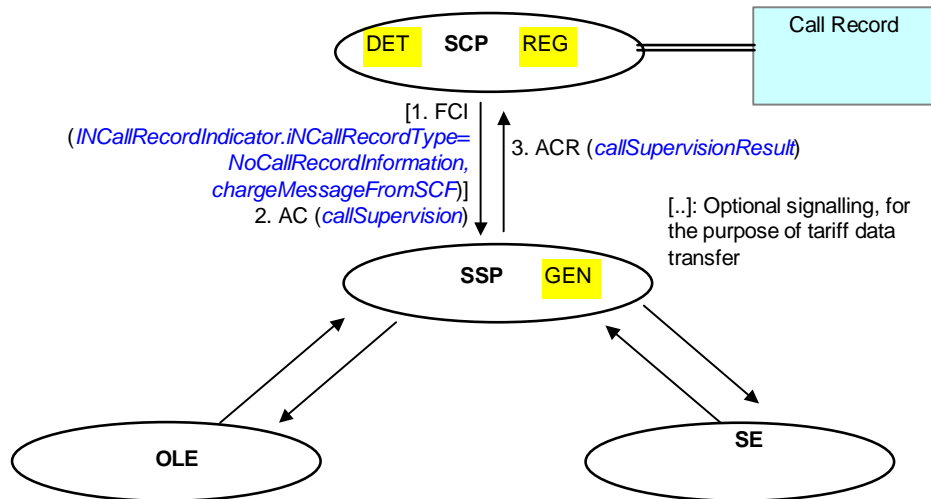

Figure 66: Scenario B.2

16.3.3   Framework for the charging operations in INAP

In table 41 the framework for the charging operations in INAP relevant to the charging scenarios is depicted. Note, that neither the octet string parameters AC_BCC, CallResult, FCI_BCC and SCI_BCC are listed nor the operation addressing parameters as e.g. partyToCharge or chargingAddress.

Table 41: Charging Operations and valid charging scenarios

Scenario	DET	GEN	REG	INAP Charging Operations	Parameters
A.1	OLE	OLE	OLE	No	No
A.2	SCF	OLE	OLE	SCI note 1 [RNC, ENC] note 3	tariffMessage.crgt note 2 or.aocrg note 2, chargingControl, [eventTypeTariff, eventSpecificInformationTariff] note 3
B.1	SCF	SSF [SCF] note 4	SSF	FCI	chargeMessageFromSCF, freeFormatData, partyToChargeIdentifier, INCallRecordIndicator.iNCallRecordType = GenerateCallRecord or AppendDataToCallRecord or OverwriteDataForCallRecord or NoCallRecordInformation (if used to update charging tariff information or to send add-on charging information), hotBillingRequired
B.2	SCF	SSF, [SCF] note 5	SCF	AC, ACR, [FCI] note 6	callSupervision, callSupervisionResult with sub-parameter ACChargingAddress/= bNCF and equal to parameter chargingAddress in FCI chargeMessageFromSCF, INCallRecordIndicator.iNCallRecordType = NoCallRecordInformation with sub-parameter chargingAddress equal to ACChargingAddress parameter in AC
B.3	SCF	SCF	SCF	No	no
NOTE 1:	Instead of the SCI also the FCI operation may be used to transfer service specific call data into the PSTN standard call record. This is a network operator specific option.					
NOTE 2:	The "charging control indicator" information has to be included in the crgt or aocrg parameter. Value "subscriber charging" is provided if the charging information shall be used for scenario A.2.					
NOTE 3:	Optional, for the purpose of SCP involved/controlled charge acknowledge supervision.					
NOTE 4:	The SCF may collect additional data for GEN function and sends them using freeFormatData to the SSF for registration.					
NOTE 5:	The SCF may add additional charges.					
NOTE 6:	Is used for Tariff Data transfer.					

16.4   On-line Charge Information provision to the user access (ONC)

With respect to the ONC function, some special aspects have to be considered. This function is always located at the local exchange (e.g. OLE) and usually closely related to the BNCF (scenario A).

If the ONC is completely handled in the PSTN (**scenario C.1**) no special charge information is provided from the IN for the feature. In general the same charges as used for scenarios A are displayed to the user's access without any additional IN intervention.

In the other case (**scenario C.2**) it relates to one of the charging scenario's 'B'.

ONC is shared between IN and PSTN/PLMN. In this case the SCF controls the charge information to be displayed to the subscriber directly.

Therefore the SCF instructs the SSF on charge information to be forwarded to the subscriber's display.
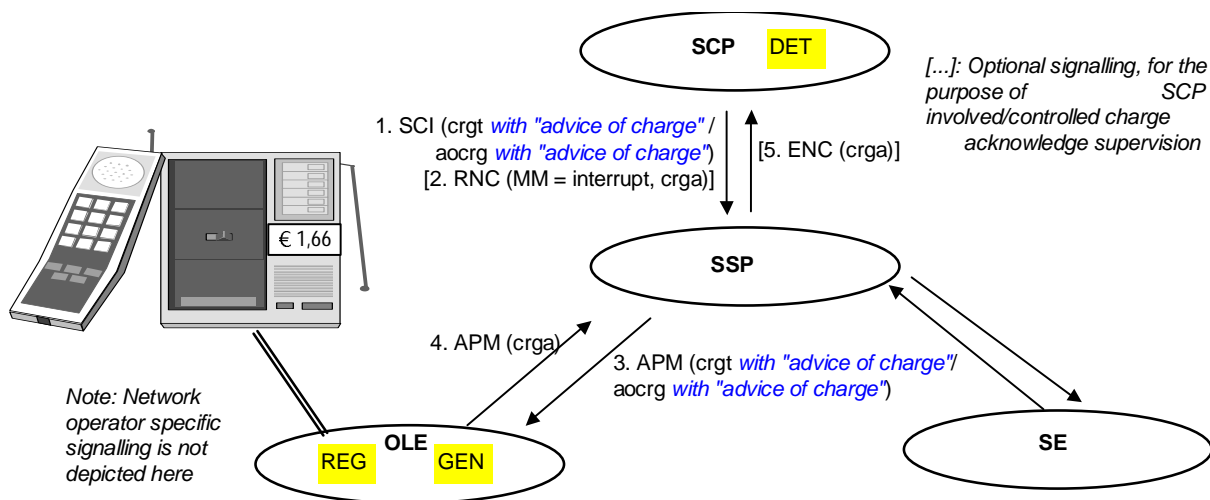
Figure 67: Scenario C.2

It is up to the Network Provider to specify details of the interworking, that means the mapping of this INAP/ISUP charging operations and parameters on the user/network interface signalling (e.g. DSS1).

In table 42 the framework for the charging operations in INAP relevant to the ONC scenarios is depicted.

Table 42: Charging Operations and ONC charging scenarios

	DET	GEN	REG	INAP Charging Operations	Parameters
C.1	OLE	OLE	OLE	No	no
C.2	SCF	OLE	[OLE] note 1	SCI [RNC, ENC] note 3	tariffMessage.crgt note 2 or.aocrg note 2, [eventTypeTariff, eventSpecificInformationTariff] note 3
NOTE 1:	Registration of ONC-charges may be network operator specific and can be independent of the access-related charges.
NOTE 2:	The "charging control indicator" information has to be included in the crgt or aocrg parameter. Value "advise of charge" is provided if the charging information shall be used for scenario C.2.
NOTE 3:	Optional, for the purpose of SCP involved/controlled charge acknowledge supervision.

16.5　Call Supervision

Call supervision means the function to supervise a limit which may be expressed in terms of time or currency or pulses. Call supervision assumes always the execution of on-line charging and is obviously to be performed at the charge generation point. The result(s) of the call supervision may be used for different applications, e.g. pre-paid card services, credit card services or to inform the service user about the used charges via announcements.

From the INAP point of view only cases are of interest, were the supervision itself and the registration or evaluation of the result is done in different nodes. The latter could be seen as supervision control function and is always performed at the SCP.

Only these type of call supervision is regarded here.

Consequently call supervision may relate to both types of scenario A and to the types B.1 and B.2. Regarding scenario A the Basic Network Charging Function must be located at the SSP (e.g. SSP in local exchanges).

In table 43 the framework for the charging operations in INAP relevant to the call supervision is depicted.

Table 43: Charging Operations and call supervision charging scenarios

Charging scenario supervised	Parameters of AC, ACR
A.1	callSupervision with and callSupervisionResult sub-parameter ACChargingAddress = bNCF note 1
A.2	callSupervision and callSupervisionResult with sub-parameter ACChargingAddress = bNCF note 1
B.1	callSupervision and callSupervisionResult with sub-parameter ACChargingAddress/= bNCF and equal to parameter chargingAddress in FCI
B.2	same as listed in table 41, scenario B.2 note 2
NOTE 1:	This is only applicable, if the SSP is located at the GEN point of the BNCF.
NOTE 2:	The used charges as reported via ACR are used for both call supervision control function and charge data registration.

16.6　Interworking with other charge determination points

16.6.1　Complete SCP control

In this option SCF has control of the charging information and instructs SSF to monitor and intercept the charge related signalling message(s) received from a higher exchange (this may be e.g. other SSF-SCF instance or an international gateway). This implies, that Charge Data Determination functions are shared between higher exchange and SCP.

Based on criteria supplied by the SCF, the SSF will send this information to the SCF immediately after receipt of the appropriate message type. Subsequently, the SCF may use this information performing its charging control (use the received charging information in the call record generation or to adjust new charge rates/pulses/charging tariff information/add-on charging information to be sent to the SSF for further processing).

In this case the operation RNC has to be used for requesting the report and interception of charging events (via parameter monitorMode = Interrupted).

Receiving one of these messages the SSF will immediately forward it (without any processing) to the SCF via operation ENC. The SCF - the receiver of the charge messages - is responsible for acknowledging these charge messages. Otherwise the sender would possibly clear the whole call because of expiration of the supervision timer of the charging tariff information or add-on charging information. For this acknowledgement the SCI operation with parameter crga must be sent.
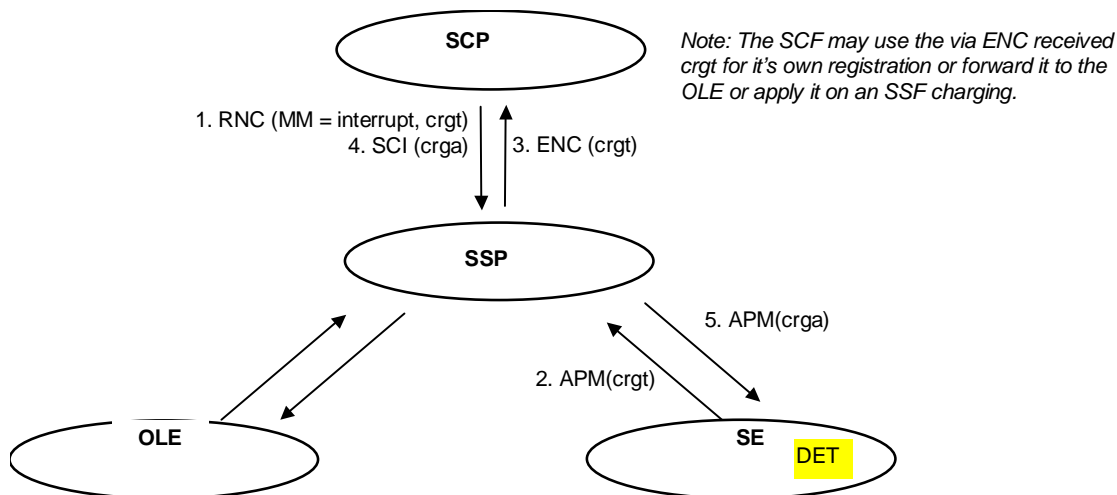

Figure 68: Complete SCP control of crgt

16.6.2   SSP processing

16.6.2.1   General

SSP processing is the application of the functions of the charge generation/registration point or the application of the functions of the connection control point as defined within the ES 201 296. In the following a distinction is made between charge messages which relate to subscriber charge and charge messages which relate to the On-line Charge Information provision (ONC). Because instead of the term ONC the term 'advice of charge' is used within the ES 201 296 both terms are used with the same meaning in the following.

   NOTE: In case of networks which support operator specific charge messages on one hand not all of the described functions will be applicable and on the other hand there may be additional capabilities supported. However, also for this charge messages the handling shall base on the same principles as described here.

The application of the functions of the charge generation/registration point in the context of 'SSP processing' is always related to one of the scenarios B.1 or B.2, because at least the charge data generation has to be located at the SSP. Only subscriber charge messages are relevant here.

In case charge messages have to pass a SSP this SSP has to act as a Connection Control Point according to ES 201 296. Connection control point function (CCPF) is related either to charging scenario A.2 for the passing on of charge messages to the Basic Network Charging Function (BNCF) or to ONC scenario C.2 for passing on charge messages to the On-line Charge Information provision to the user access (ONC). For scenario A.2 only subscriber charge messages are relevant. For scenario C.2 only advice of charge messages are relevant.

A special case is the application of subscriber charge messages on the charge generation/registration at the SSP and the passing on of these charge messages as advise of charge messages to the ONC function. This special case can be seen as a combination of the functions of the charge generation/registration point at the SSP and the functions of the connection control point for passing on charge messages to the ONC function.

The SCF has to instruct the SSF which charging application shall process the charge messages. Internally the SSP distributes the charge message received to the correct charging application. This distribution is depicted in figure 69. It depends on what the SCP has specified in the FCI or the SCI operation, and what is the default handling in the SSP (either no SCI or/and no FCI received in SSP).
Details of the INAP control via operations and parameters are listed in tables 41 and 42.
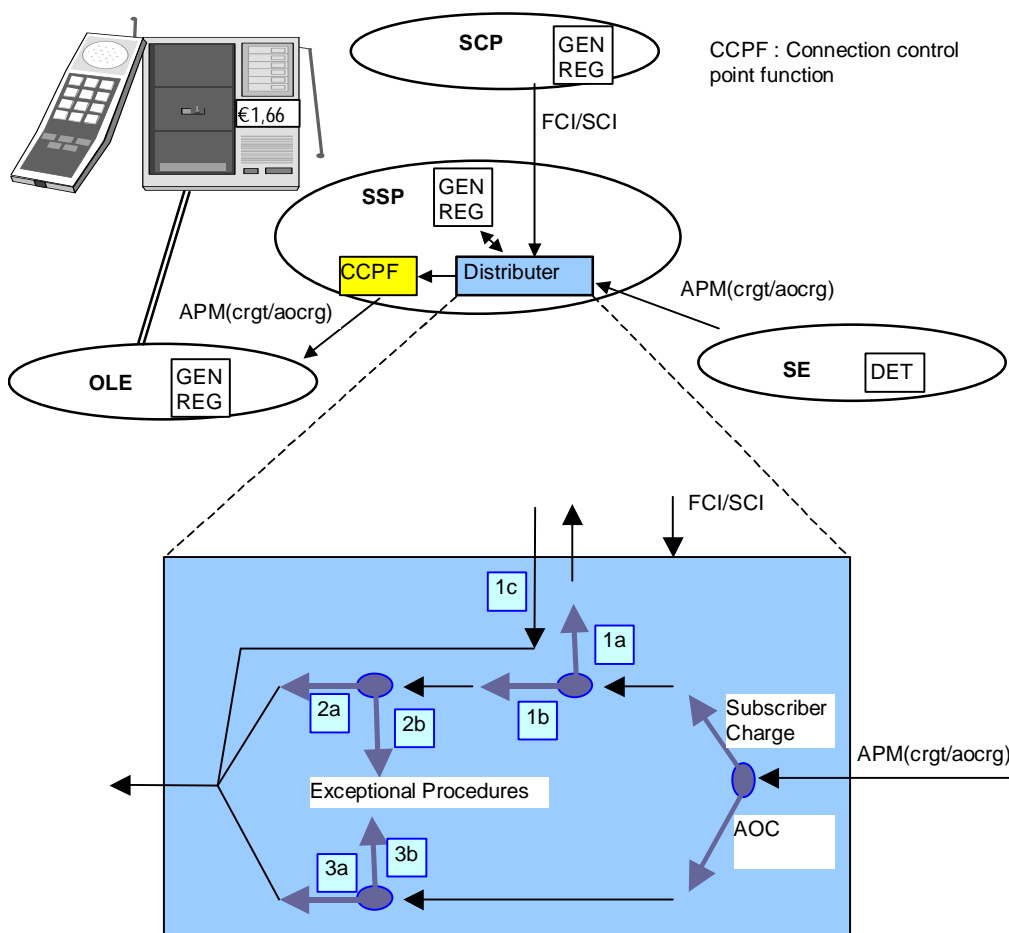
Figure 69: Distribution of charge messages for SSP processing

Table 44: Handling of charge messages that relate to 'subscriber charge's

No. in figure 69	Controlling INAP Operation	Parameters	Handling at SSP
1a	FCI	TariffFromSuccExchange = takeIntoAccountNoAOC or TariffFromSuccExchange = takeIntoAccountTranslateIntoAOC	The charge message is passed to the GEN and/or REG function at the SSP. There the procedures of the application process of the Charge Generation/Registration Point are performed as defined within the ES 201 296.
1b	FCI	TariffFromSuccExchange = notTakenIntoAccount or Parameter TariffFromSuccExchange not present or no FCI received	The charge message is not passed to the GEN and/or REG function at the SSP. Distribution not yet finalized.
1c	FCI	TariffFromSuccExchange = takeIntoAccountTranslateIntoAOC	After validation the translated charge message (with indication 'Advice of Charge' instead of 'Subscriber Charge') is passed on to the Connection Control Point Function of the SSP.
2a	SCI	as shown in table yyy of the SCI procedure for the 'relay' of subscriber charge messages	The charge message is passed to the Connection Control Point Function of the SSP. The Connection Control Point Function is performed as defined within the ES 201 296.
2b	SCI	as shown in table yyy of the SCI procedure for the 'no relay' of subscriber charge messages or no SCI received	The charge message is discarded at the SSP. Exceptional procedures as defined within the ES 201 296 for Charge Generation/Registration Point are invoked.

**Table 45: Handling of charge messages that relate to 'advice of charge'**

No. in figure 69	Controlling INAP Operation	Parameters	Handling at SSP
3a	SCI	RelayAOCfromDestination = TRUE	The charge message is passed to the Connection Control Point Function of the SSP. The Connection Control Point Function is performed as defined within the ES 201 296.
3b	SCI	RelayAOCfromDestination = FALSE or Parameter RelayAOCfromDestination not present or no SCI received	The charge message is discarded at the SSP. Exceptional procedures as defined within the ES 201 296 for Charge Generation/Registration Point are invoked.

Acknowledgement of the charge messages is in the responsibility of the charging application which processes the charging tariff information or add-on charging information.

For scenario A.2 this is depicted in figure 70 and for scenario B.1 an example is depicted in figure 71 and for scenario B.2 an example is depicted in figure 72.

If the charge message is neither to be relayed to BNCF for scenario A nor to be applied of IN specific charge data generation for scenario B.1 or B.2 no positive charge acknowledge message must be generated.

The SCF is not responsible for acknowledging these charge messages.
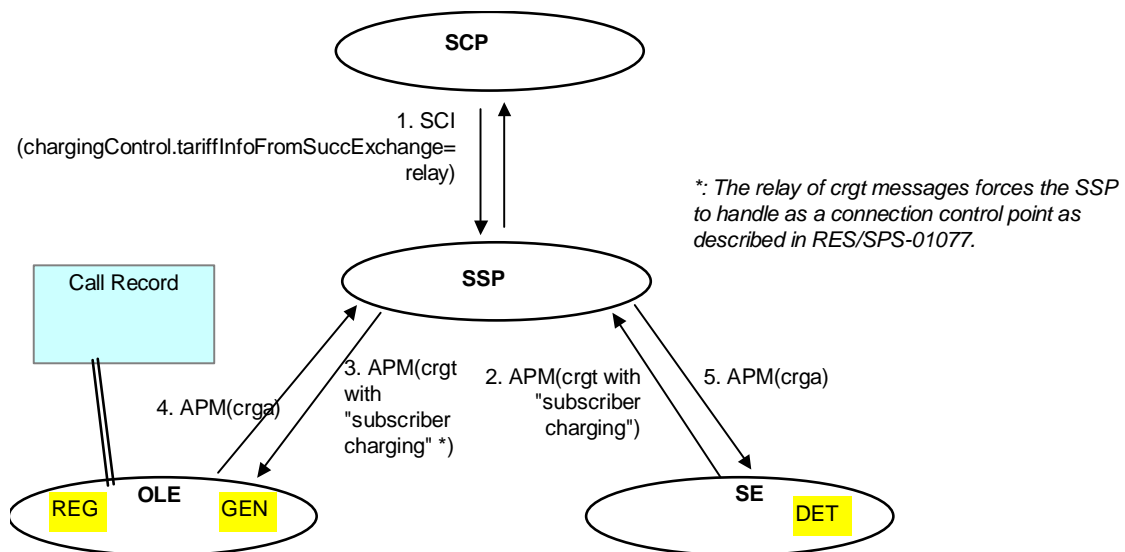
Figure 70: Relay of crgt to BNCF in scenario A.2
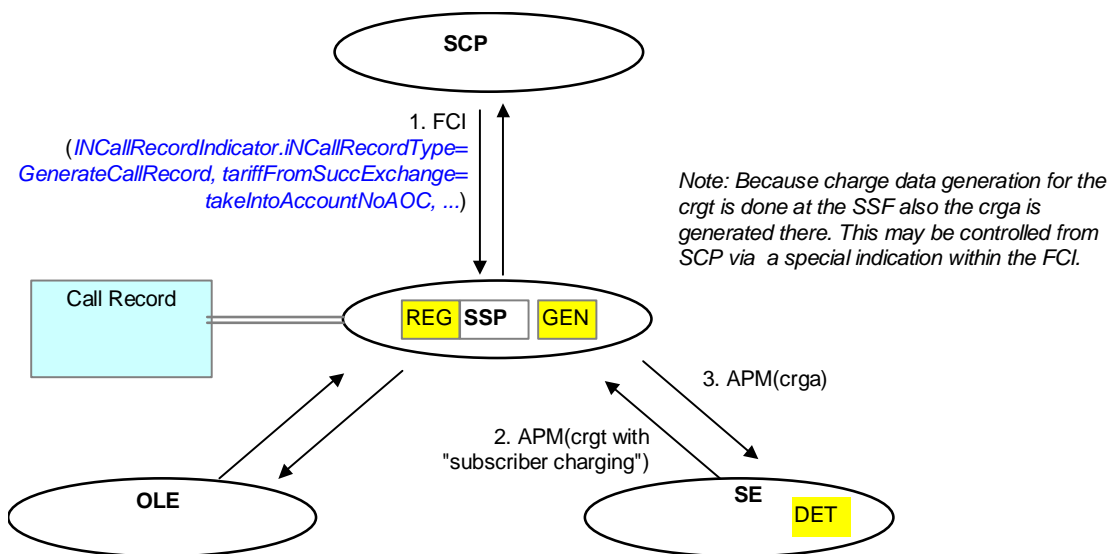
Figure 71: Direct processing of crgt in scenario B.1
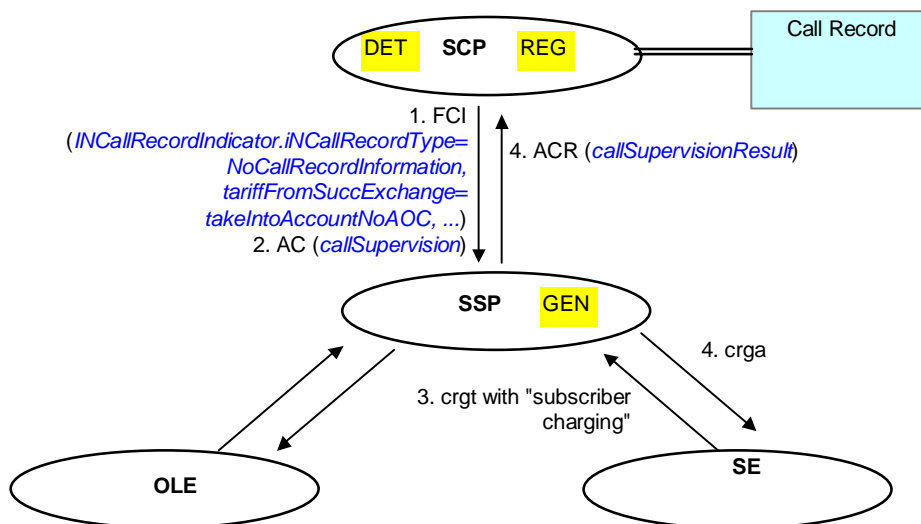
Figure 72: Direct processing of crgt in scenario B.2

16.6.2.2   SCP Monitoring

The SCF is not involved in Charge Determination and just monitors the charging events initiated from a higher exchange.

- The operation **RNC** has to be used for requesting the report of charging events (via parameter *monitorMode = NotifyAndContinue*).

- Receiving one of these messages (from B-side or an other IN service logic) the SSF will forward it to the SCF via operation **ENC** immediately. Processing of the charge messages is done as described above in clause "SSP Processing".

- The SCF is not responsible for acknowledgement of these charge messages.

Figure 73: Combination of direct processing of crgt in scenario B.1 with SCP Monitoring of crgt
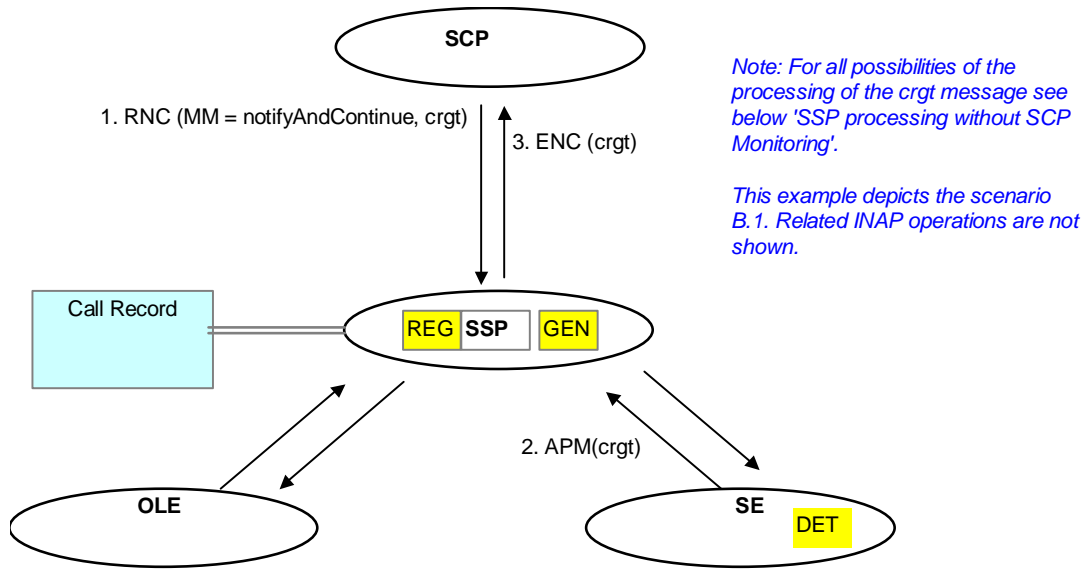
Annex A (informative):
SDL Diagrams for IN CS-3

The SDL diagrams for IN CS-3 Call Party Handling are available in an electronic attachment accompanying
EN 301 931-4.

The SDL diagrams are informative, and serve to illustrate the operation of call party handling.
The SDLs follow the model structure of the SSF-FSM as described herein. The SDL uses an object oriented approach,
with separate instances of the CSA, CS, SSF-CCF, O-BCSM and T-BCSM.

The SDLs are a full data model, and therefore can be used for simulation purposes.
The ASN.1 used in the SDL model is taken directly from the present document. The SDLs are based on the SDLs
produced for IN CS-2, corrected and updated with IN CS-3 behaviour. They have been extensively simulated to verify
their correct behaviour.

There are however, some limitations with the SDL model. The model only concerns the SCF-SSF interface. The model
is intended to cover call party handling only, it is not a full model of the SCF-SSF interface. It is intended to illustrate
the CSCV transitions on signalling events, Detection Points and SCF operations. Where the model does go beyond
CPH, in making reference to charging operations, etc., then these are not fully modelled, in particular if they have no
impact on CPH and on CSCV transitions.

In case of discrepancy between the main text and SDLs the main text of the present document shall take precedence.

Annex B (informative):
Call Views examples within CCF/SSF

B.1   Introduction

Multi Party capability (CPH) allows several parties to be involved within a call initiated either by an user or by an SCF.

Multi Service capability (Single Point of Control or Multi Point if Control) allows several services to be activate on the same BCSM.

Both imply that several BCSM, several signalling termination and several physical terminations coexist within the same CCF/SSF instance. Rules need to be defined regarding:

- How the signalling may be handled between all the parties and how the service is aware of events.

- How the real bearer connection may be handled between all the parties and how the service really influence it.

More detailed figures identifies major element that are useful for understanding how CCF/SSF behaves such as BCSM, FIM agent.

Definition:

Per call (CCF)

- CCFi: CCF instance « i ». created due to setup Abstract Signalling Primitive received from a calling user via a BST, or a remote O-BCSM via IBI, or a first Initiate Call Attempt within a newly CSA.

- SSFi: SSF instance « i »: created when the first service is triggered from a CCFi.

- BSTi i: Basic Signalling termination Incoming "i": signalling event.

- BSTo i: Basic Signalling termination Outgoing "i".

- DSTi i: Dummy Signalling termination Incoming "i": signalling event.

- DSTo i: Dummy Signalling termination Outgoing "i".

- T i: Termination: physical circuit or port "i".

- RTi: Remote termination "i". Logical reference between two Connection Points.

- CP: Connection Point within a termination context. A connection point is always inserted between Ti/RTj or RTi/RTj.

- IBI i-j: Inter BCSM Interface between CCFi and CCFj.

Per service

- CSA: Call segment association, global connection view for one service within a SSF instance.

- CSi: Call Segment "i".

- Legi: represents one party for a service only.

Per CCF instance

- FIMa: FIM agent that manages interaction between services triggered from the same BCSM.

- Oi: BCSM O "i".

- Ti: BCSM T "i".

B.2   General view

B.2.1   Single Two party call

Here after is a view that represents a call in Stable phase, naturally composed by an Originating Half call and Terminating Half call. Only elements known outside the CCF/SSF are represented.

- Two services on the Originating part, CSA and CS represents the connection view for each service.

Regarding the call:

- Signalling events are received/sent/relayed from/to/between BSTi 1 and BSTo 2.

- Physical connection is established between T1 and T2.
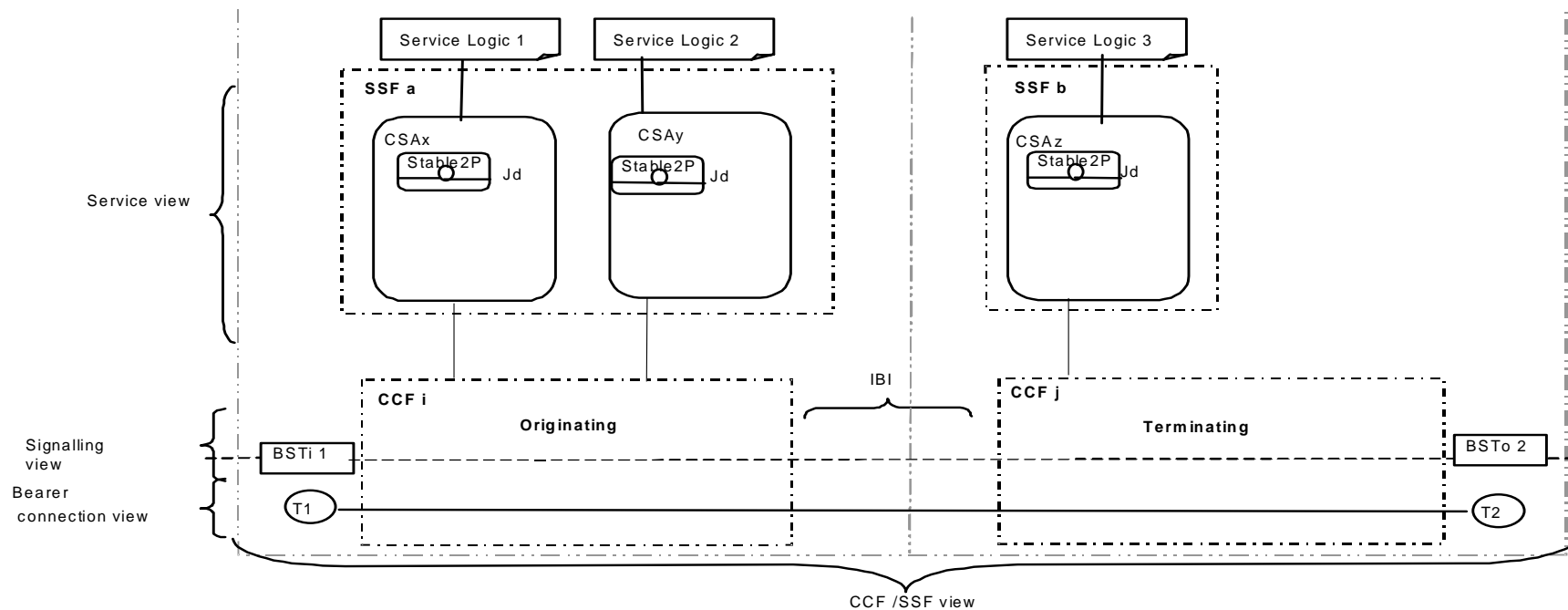
Figure B.1

B.2.2   Multi party call

Same configuration as previous one and in addition one of the service in originating side as initiated a call.

- New CCFk is created for dealing with the terminating side of the new SCF initiated call. SSFc may be created if new service on T-BCSM from CCFk. New BSTo3, new T3.

Regarding the call:

- Signalling events are received/sent/relayed from/to/between BSTi and BSTo 2.

- Signalling events from are sent are received by/sent/relayed from/to/between DSTi4 and BSTo 3. Nothing goes outside issued from DST4.

- Physical connection is established between T1 and T2.

- T3 is alone.
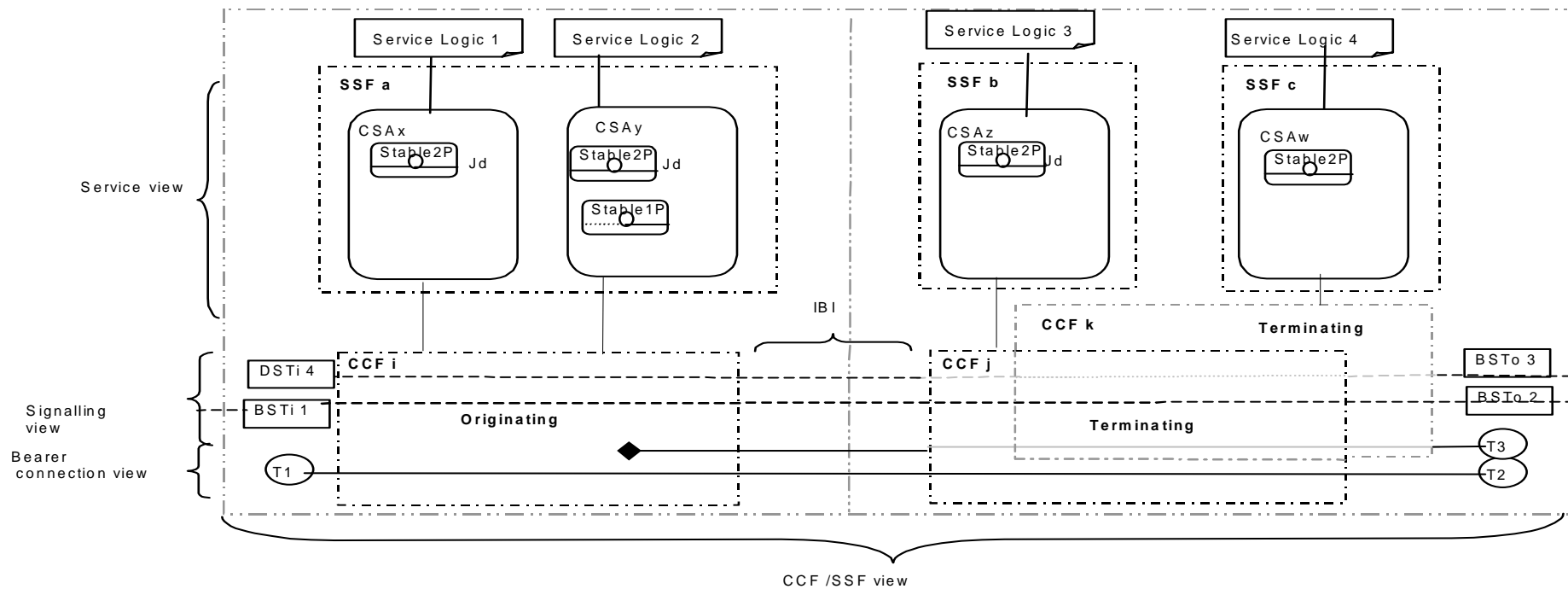
Figure B.2

B.3 Detailed view

B.3.1 Multi Party Call detailed view

Here after is a view that represents a call (BST1-BST2; T1-T2) in stable phase, naturally composed by a Originating Halfcall and Terminating Half call. Two services SL1 and SL2 on the Originating part, one in the Terminating part SL3.

SL 2 initiates a call (BST3,T3).

SL3 initiates a call (BST4,T4).
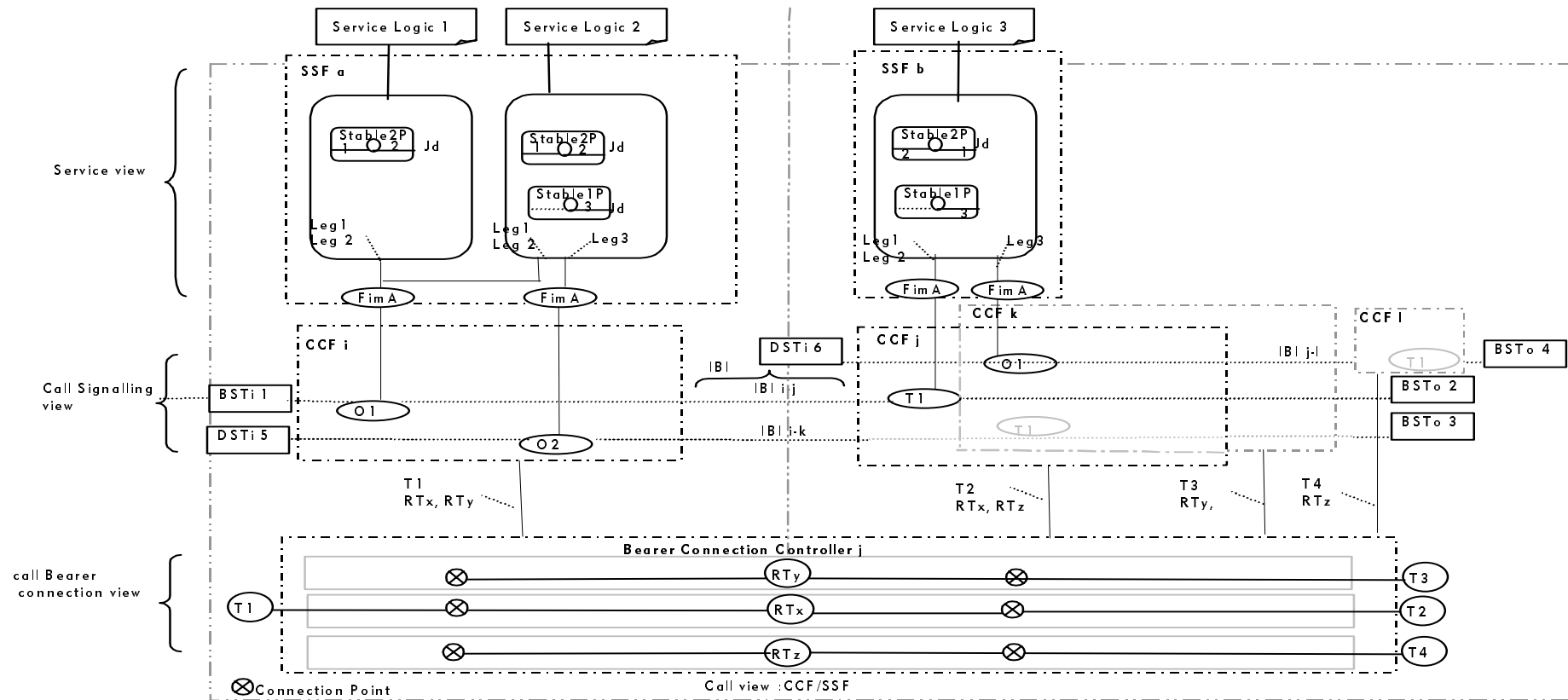
Figure B.3

Initial call

- BSTi1 ↔ BSTo2.

- T1 X T2.

SL2 initiated call

- DSTi5↔ BSTo3.

- T3.

SL2 initiated call

- DSTi6↔ BSTo4.

- T4.

B.4   Examples

B.4.1   Example 1

- From previous situation in clause B.3.1.

- Service 2 Merge the two CS.

- SL 3 and resources disappeared.
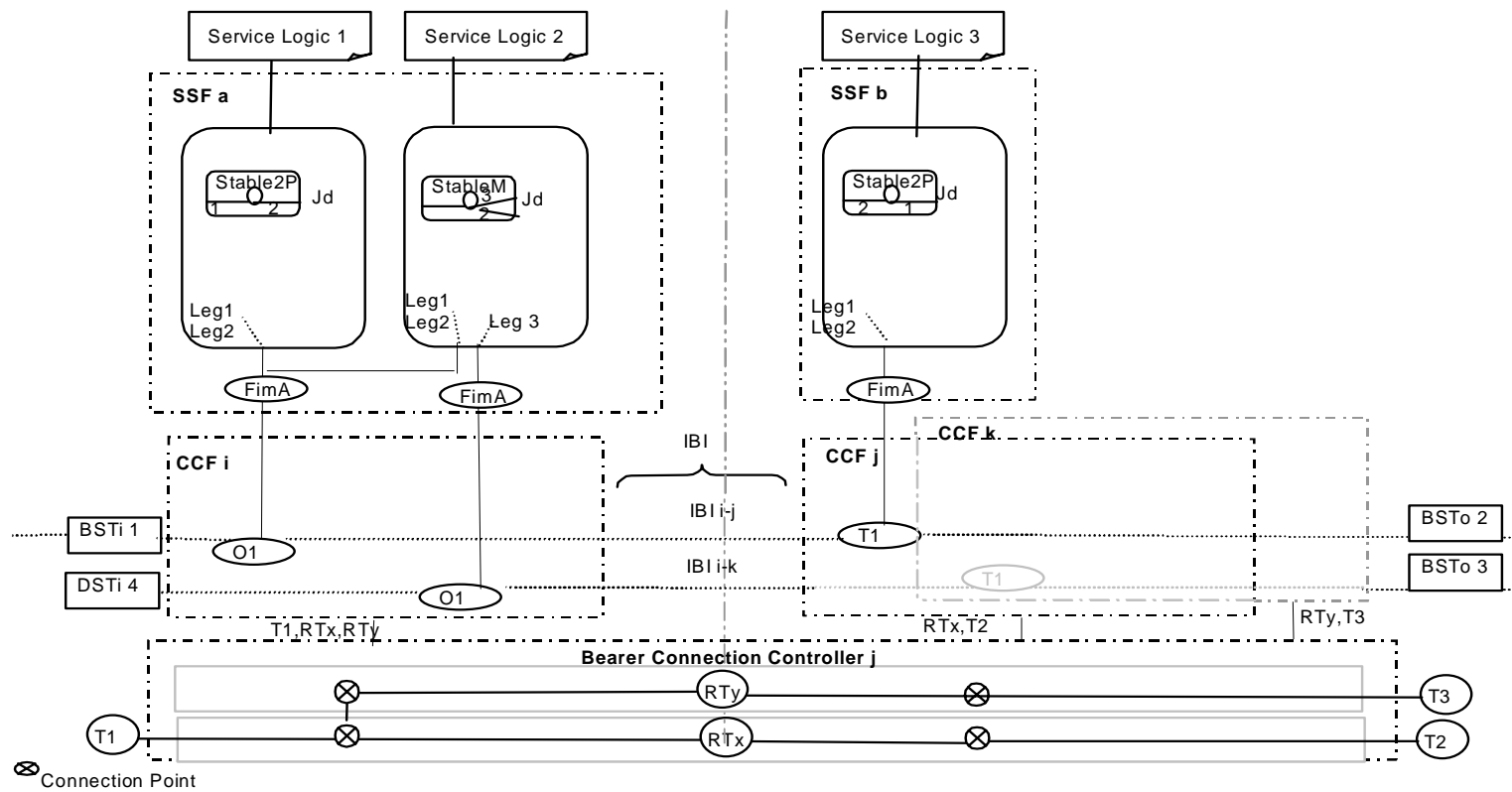
Figure B.4

Signalling path remains unchanged.

RTy is now connected to a CP thus T1/T2/T3 are through connected.

Initial call

- BSTi1 ↔ BSTo2.

- T1 X T2 X T3.

SL2 initiated call

- DSTi5↔ BSTo3.

- T1 X T2 X T3.

B.4.2   Example 2

From previous example:

- Service 3 splitleg leg 2.

- Service 2 splitleg leg 2.
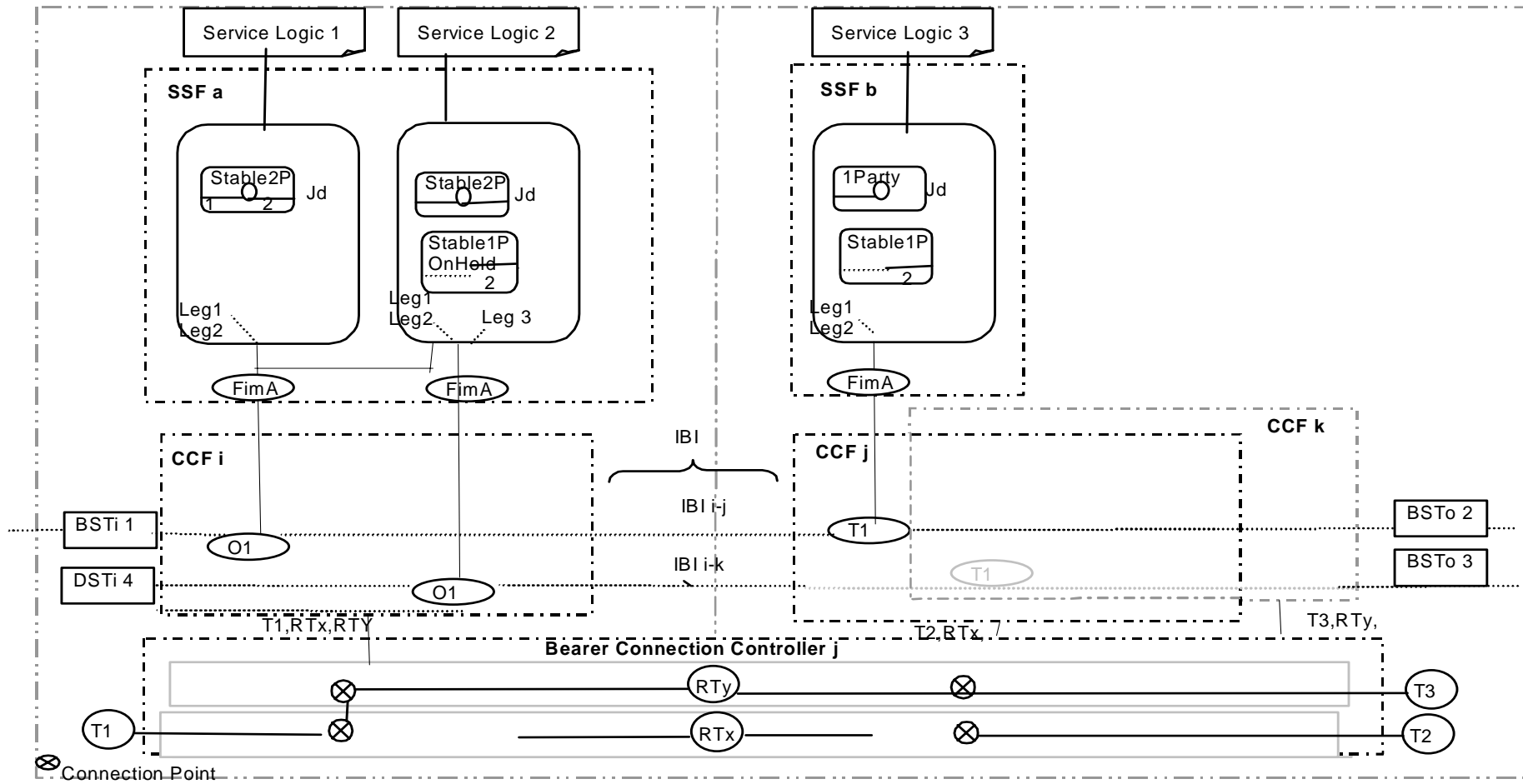
Figure B.5

T1 is connected to T3, T2 is alone

- No impact on signalling.

2sd step:

- Service 3 Merge two CS.
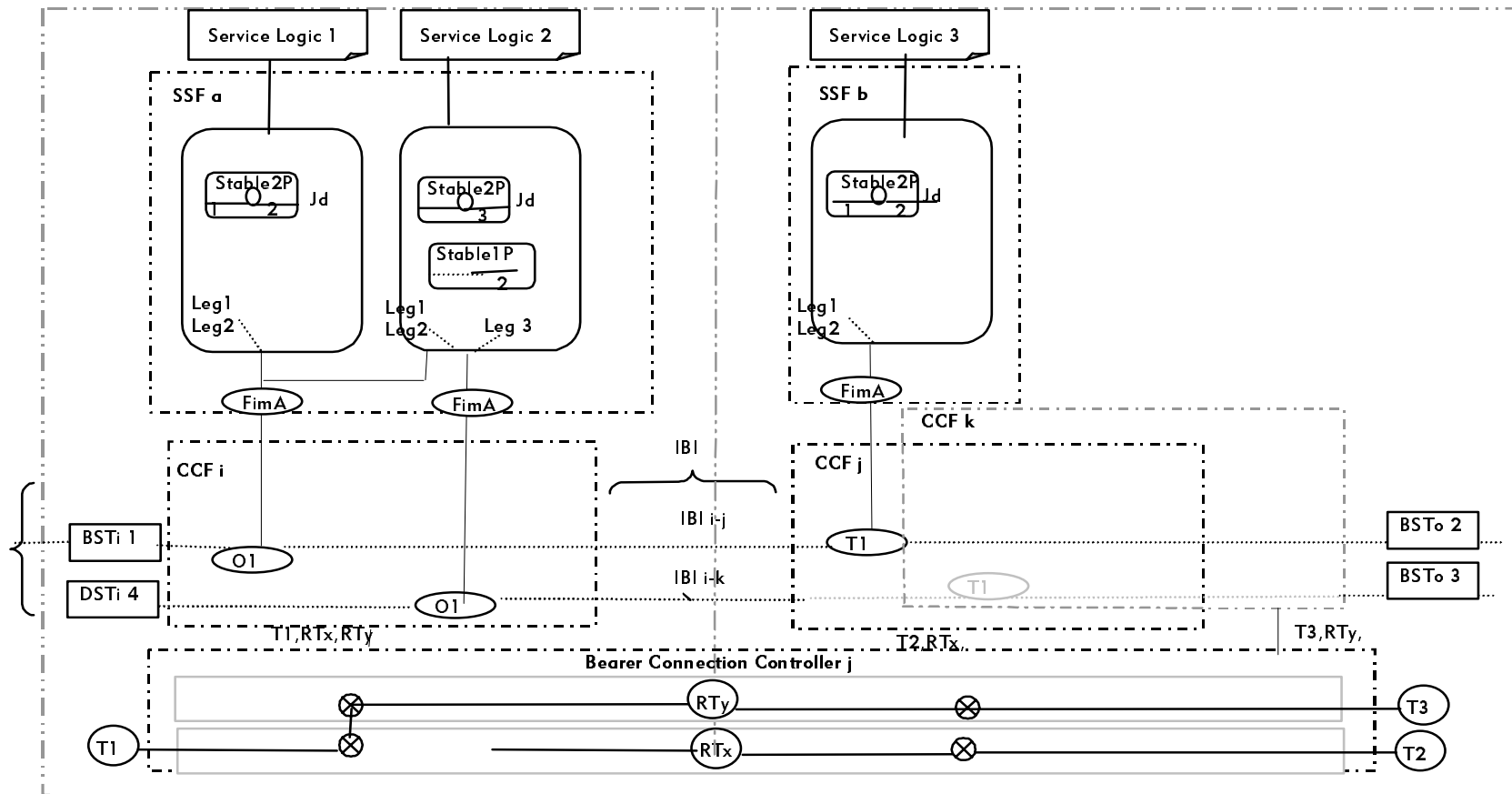
Figure B.6

T1 still be connected to T3, T2 is always alone, **it could be reconnected on MergeCs from Service Logic 2.**

- No impact on signalling.

B.4.3 Example 3

- Connect P2 on terminatingSetup and stable phase is reached.
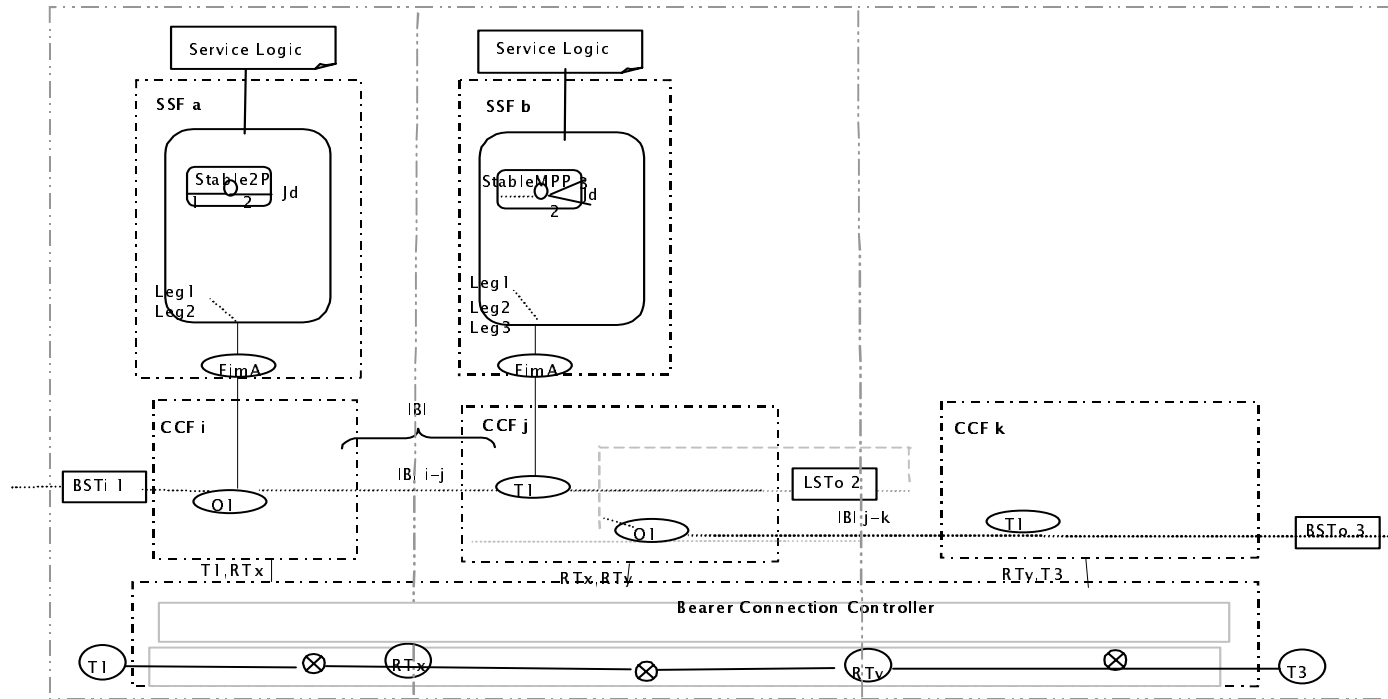
Figure B.7

Initial call:

- BSTi1 - BSTo3.

- T1 X T3t.

Annex C (informative):
Signalling Terminations, Rules and Configurations

Warning: the rules contained herein are inconsistent and are provided for information only.

C.1   Signalling Terminations

Within the CCF, Basic Signalling Terminations provide adaptation functionalities between the Basic Call Controller and signalling interfaces.

NOTE: Rules is to indicate which abstract primitives are relayed to the T_BCSM and which are discarded by the Basic Call Controller.

Dummy and Loop Signalling Terminations do not send any signalling message on external interfaces.
For the CCF different types and variants of Signalling Terminations may be distinguished:

Basic Signalling Termination: BST - two variants are envisaged.

- **Basic Relay:** BST-r
  Signalling Terminations that belong to that type do send signal and relays any received signal. Signals are transferred to this Signalling Termination and handled as they are received from the BCSM. The signals received by the Signalling Termination from one BCSM are to be relayed based on generic rules to be defined for signalling interworking.

- **Basic Filtering:** BST-f
  Signalling Terminations that belong to that type do send signal and relay received signals but there may be reduced end-to-end signalling capability. Signals are transferred to this Signalling Termination and handled as they are received from the BCSM. The signals received by the Signalling Termination from one BCSM are discarded based on generic rules to be defined for signalling interworking.

In addition to those supporting a particular signalling protocol, there are also two specific types of Signalling Terminations:

- **Dummy:** DST
  Signalling Terminations that belong to that type do not send any signal and stores any received signal. When a BCSM becomes connected again to a Signalling Termination, the stored signals are transferred to this Signalling Termination and processed as if they had been received from a BCSM.

- **Loop:** LST
  An instance of this type is created when an O_BCSM instance is created as a result of a Connect operation. Both the T_BCSM and the newly created O_BCSM are associated with the Signalling Termination. The signals received by the Signalling Termination are relayed to the Basic Call Controller Process that may pass them to the T_BCSM instance.

LST has two variants:

- **Loop Relay:** LST-r
  The signals received by the Signalling Termination from one BCSM are relayed based on generic rules to be defined for signalling interworking.

- **Loop Filter:** LST-r
  The signals received by the Signalling Termination from one BCSM are discarded based on generic rules to be defined for signalling interworking.

Dummy and Loop Signalling Terminations do not send any signalling message on external interfaces.

C.2   Signalling Interworking Rules

The Signalling Terminations and the BCSM instances may enforce the following rules when processing signalling information:

- R0: Abstract Signalling Primitives received by Signalling Terminations shall be discarded or mapped to an appropriate signalling message, based on the type of signalling system and on the states of the signalling FSM, i.e. the "FSM for Call Control Signalling Termination" as defined in clause 6.

- R1: Abstract Signalling Primitives received from a Signalling Termination by a BCSM instance are converted into the appropriate IBI Abstract Signalling Primitives and sent to the peer BCSM instance.

- R2: In case there is no peer BCSM instance, Abstract Signalling Primitives are buffered in the BCSM until it becomes connected to a peer BCSM. However, in case the information contained in the Abstract Signalling Primitive is not essential for the call, it may be discarded.

- R3: Abstract Signalling Primitives received from the IBI by a BCSM instance are converted into the appropriate Abstract Signalling Primitives and sent to its associated Signalling Termination.

- R4: Abstract Signalling Primitives received from a BCSM instance by a Loop Signalling Termination (LST) are discarded if no signalling transparency applies, otherwise they may be relayed.
  They are treated as follows:

Table C.1: Handling of Abstract Signalling Primitives on a CSCV "Stable_Multi_Passive_Party" call

Abstract Signalling Primitive	Action in loop termination Signalling transparency case	Action in loop termination No signalling transparency case
Setup Req	NA	NA
Setup Resp	Relayed to incoming leg and mapped to a signalling message, if Setup confirmation not yet sent else discarded (note 1)	Discarded (notes1 and 2)
SubsequentAddress	NA	NA
CallProgressReq	May be relayed and mapped to a signalling message	May be relayed and mapped to a signalling message
NetworkSuspend	Discarded or relayed to incoming leg and mapped to a signalling message (note 3)	Discarded
NetworkResume	Discarded or relayed to incoming leg and mapped to a signalling message (note 3)	Discarded
ServiceFeatureIndication	Discarded	Discarded
ChargingEvent	May be relayed and mapped to a signalling message	Discarded
Data	May be relayed and mapped to a signalling message	Discarded
Release	See Release Handling Rules	See Release Handling Rules
NOTE 1: A CallProgress(Progress)Req Abstract Signalling Primitive may be generated and mapped to a signalling message.		
NOTE 2: If Setup confirmation (answer) has not been sent to calling party a Setup confirmation may be conveyed to incoming leg and mapped to a signalling message.		
NOTE 3: Refer to e.g. EN 301 070-1.		

- R5: In case more than one BCSM instance is associated with the same Basic Signalling Termination (BST), Abstract Signalling Primitives received from one BCSM instance are handled as follow:

Table C.2: Handling of Abstract Signalling Primitives for a CSCV "Stable_Multi_Party" call

Abstract Signalling Primitive	Action Signalling transparency case	Action No signalling transparency case
Setup Req	NA	NA
Setup Resp	Relayed to incoming leg and mapped to a signalling message if Setup confirmation not yet sent else discarded (note 1)	Discarded (notes 1 and 2)
SubsequentAddress	NA	NA
CallProgressReq	May be relayed and mapped to a signalling message	May be relayed and mapped to a signalling message
NetworkSuspend	Discarded or relayed to incoming leg and mapped to a signalling message (note 3)	Discarded
NetworkResume	Discarded or relayed to incoming leg and mapped to a signalling message (note 3)	Discarded
ServiceFeatureIndication	Discarded	Discarded
ChargingEvent	May be relayed and mapped to a signalling message	Discarded
Data	May be relayed and mapped to a signalling message	Discarded
Release	See Release Handling Rules	See Release Handling rules
NOTE 1: A CallProgress(Progress)Req Abstract Signalling Primitive may be generated and mapped to a signalling message.		
NOTE 2: If Setup confirmation (answer) has not been sent to calling party a Setup confirmation may be conveyed to incoming leg and mapped to a signalling message.		
NOTE 3: Refer to e.g. EN 301 070-1.		

- R6: When call processing is suspended, all Abstract Signalling Primitives received from a Signalling Termination or the IBI are blocked (i.e. not conveyed) until call processing is resumed, except for Data and Charging Abstract Signalling Primitives. However, the transfer of Data and Charging Abstract Signalling Primitives may be blocked as a result of SCF instructions (e.g. RequestReportUTSI).

C.3   Examples: Relationships between CSCV states and signalling configurations
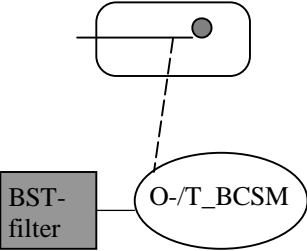
CSCV state	Signalling Configuration	Graphical representation
		▬▬▬ = signalling transparency end-to-end ———— = no signalling transparency end-to-end
Null: This state represents the condition where no call processing is active and there is no controlling leg or passive leg connected to the connection point.	**SC0:** No BCSM instance, no Signalling Termination. This configuration represents an idle call.	
Originating_Setup: This state represents a call segment instance with a joined controlling leg and a pending passive leg with an associated O_BCSM (e.g. an originating two-party call in the setup phase).	**SC1:** This configuration represents an O-BCSM instance associated with one Basic Signalling Termination that provides the interface to the incoming Signalling Path., but no peer BCSM instance. Entry events: A – Receipt of a signalling event indicating a desire to place an outgoing call. Exit events: – To SC2: Creation of the peer T-BCSM.	1

Originating_Setup:	**SC2:**	
The establishment of a basic two-party call A to B, i.e. the original setup of a two party call.	**2a**	
	SC2a: This configuration represents an O-BCSM associated with a Basic Signalling Termination that provides the interface to the incoming Signalling Path.	
This state represents a call segment instance with a joined controlling leg and a pending passive leg with an associated O_BCSM (e.g. an originating two-party call in the setup phase).	Entry events: - See exit events from SC1. Exit events: - To SC1: Release.Resp.Conf sent to the peer T-BCSM. - To SC12: Instruction resulting from a MergeCallSegments operation.	
	A Peer T-BCSM is created when the O-BCSM reaches the SendCall PIC. The T_BCSM becomes associated with a Signalling Termination when the FacilitySelectedAndAvailable DP is reached. This Signalling Termination provides the interface to the outgoing Signalling Path.	A BST-relay — O-BCSM — T-BCSM — BST-relay B

Originating_Setup:	SC2b:	2b
AFTER connect –I N CALL to B2, i.e. a follow on call. This state represents a call segment instance with a joined controlling leg and a pending passive leg with an associated O_BCSM (i.e. a follow on call).	This configuration represents an O-BCSM associated with a Basic Signalling Termination tl provides the interface to the incoming Signalling Path. Entry events: - See exit events from SC1. Exit events: - To SC1: Release.Resp.Conf sent to the peer T-BCSM. - To SC12: Instruction resulting from a MergeCallSegments operation. Filtering is applied in the BST (follow-on call to B2). Signalling transparency for e.g. APM. A Peer T-BCSM is created when the O-BCSM reaches the SendCall PIC. The T_BCSM becomes associated with a Signalling Termination when the FacilitySelectedAndAvailable DP is reached. This Signalling Termination provides the interface to the outgoing Signalling Path.	 A
Terminating_Setup:	SC3:	3
After FacilitySelectedAndAvailable DP is reached **i.e. after Setup.Req has been sent**): This state represents a call segment instance with a pending controlling leg and a joined passive leg with an associated T_BCSM (e.g. a terminating two-party call in the setup phase).	This configuration represents a T-BCSM (connected to a peer O-BCSM) with a Basic Signalling Termination. Entry events: - See exit events from SC4. - O-BCSM reaches the SendCall PIC.	A

	Exit events: - To SC4: The T-BCSM transits to FacilitySelectedAndAvailable DP. - To SC11: Instruction resulting from a Connect operation. One T-BCSM instance that becomes associated with a Basic Signalling Termination. One peer O-BCSM instance associated with one Basic Signalling Termination, providing the interface to the incoming Signalling Path. This BCSM instance is not visible in the CSCV. Signalling transparency end-to-end, i.e. relay capability for the BST.	
Stable_2_Party: (user initiated call – relay case) This state represents a stable two-party call, and is either an originating or a terminating call from the perspective of the controlling user with a joined controlling leg and a "joined" passive leg with either an O_BCSM or a T_BCSM associated with it.	**SC4:** This configuration represents a T-BCSM (connected to a peer O-BCSM) with a Basic Signalling Termination. Entry events: - See exit event from SC2 and SC3. Exit events: - To SC3: The T-BCSM transits to the T-Busy, T-NoAnswer. - To SC13: Instruction resulting from a MergeCallSegment operation. **SC4a:** One O-BCSM instance associated with one Basic Signalling Termination that provides the interface to the incoming Signalling Path. A peer T-BCSM instance associated with one Basic Signalling Termination that provides the interface to the outgoing Signalling Path. Signalling transparency end-to-end, i.e. relay capability for the BST.	4a

Stable_2_Party: (user initiated call – no relay, filtering case e.g. add-on additional leg (using ICA)). This state represents a stable two-party call, and is either an originating or a terminating call from the perspective of the controlling user with a joined controlling leg and a "joined" passive leg with either an O_BCSM or a T_BCSM associated with it.	**SC4b:** One O-BCSM instance associated with one Basic Signalling Termination that provides the interface to the outgoing Signalling Path. A peer T-BCSM instance associated with one Basic Signalling Termination that provides the interface to the incoming Signalling Path. No signalling transparency end-to-end, i.e. filter capability for the BST.	4b Or BST-filter — O-BCSM — T-BCSM — BST-filter A B
1-Party: (relay case – original passive leg still exists e.g. after SplitLeg or MoveLeg) This state represents a 1-party call with a joined controlling leg status and no passive legs. The O_BCSM is associated with the controlling leg since no passive leg is connected to the CP.	**SC5:** This configuration represents an O-BCSM instance associated with a Basic Signalling Termination with signalling relay capability to a peer T-BCSM (i.e. signalling transparency between A-party and B-party is maintained) and another O-BCSM instance associated with a Dummy Signalling Termination (DST) with no peer T-BCSM. (This DST allows to establish an additional call from A-party to a C-party with no signalling transparency using the Connect operation while signalling transparency (end-to-end) is maintained on the call A to B). Notice that bearer connection is broken between A and B, but may be established between A-C after Connect). Entry events: - Creation of an 1-Party call with O-BCSM as the result of e.g. SplitLeg or MoveLeg operation. - See exit events from SC4.	5 BST relay — O-BCSM — T-BCSM — BST-relay A B DST — O-BCSM

	Exit events:	
	- To SC0: Release.Resp.Conf sent by O-BCSM.	
	- To SC12: Instruction resulting from a MergeCallSegments or MoveLeg.	
	One O-BCSM instance associated with one Basic Signalling Termination that provides the interface to the incoming Signalling Path.	
	A peer T-BCSM instance associated with one Basic Signalling Termination that provides the interface to the outgoing Signalling Path.	
	Signalling transparency end-to-end, i.e. relay capability for the BST.	
	One O-BCSM instance is created and associated with a Dummy Signalling Termination for the joined controlling leg.	
	In this case the signalling connection is unaffected but the bearer connection is broken for the A-party.	
1-Party: (no relay case – e.g. after DisconnectLeg (p) This state represents a 1-party call with a joined controlling leg status and no passive legs. The O_BCSM is associated with the controlling leg since no passive leg is connected to the CP.	**SC6:** This configuration represents an O-BCSM instance associated with a Basic Signalling Termination with signalling relay capability and no peer O-BCSM. A filtering capability will apply for a possible follow on call as confirmation (answer) has been sent backward to A-party. A/B Entry events: - Creation of an 1-Party call with O-BCSM or a T-BCSM as the result of e.g. DisconnectLeg operation. - See exit events from SC4.	6

	Exit events: - To SC0: Release.Resp.Conf sent by T-BCSM. - To SC12: Instruction resulting from a MergeCallSegments or MoveLeg. One O-BCSM or T-BCSM instance associated with one Signalling Termination that provides the interface to the incoming/outgoing Signalling Path. Abstract Signalling Primitives received from the Signalling Termination are stored in the BCSM until its becomes associated with a peer BCSM. No peer BCSM.	
Originating_1_Party Setup: This state represents a 1-party call with a surrogate controlling leg and a pending passive leg with which an O_BCSM is associated. This state is entered when using the ICA operation to initiate a call from SCF. No peer BCSM can exist before a continue operation has been sent to resume cal processing.	**SC7:** This configuration represents an O-BCSM instance associated with a Dummy Signalling Termination and no peer T-BCSM. Entry events: - Creation of an O-BCSM as the result of an InitiateCallAttempt operation. - See exit events from SC8. Exit events: - To SC6: The O-BCSM reaches the SendCall PIC.	7

Originating_1_Party Setup: This state represents a 1-party call with a surrogate controlling leg and a pending passive leg with which an O_BCSM is associated **.**	**SC8:** This configuration represents an O-BCSM instance associated with a Dummy Signalling Termination and connected to a peer T-BCSM.	8
	Entry events: - See exit event from SCx. Exit events: - To SC5: Release.Resp.Conf sent to the T-BCSM. - To SC7: Instruction resulting from a Connect operation. A Peer T-BCSM is created when the O-BCSM reaches the SendCall PIC. It becomes associated with a Signalling Termination when the FacilitySelectedAndAvailable DP is reached. This Signalling Termination provides the interface to the outgoing Signalling Path.	DST — O-BCSM — T-BCSM — BST — B

Stable_1_Party:	**SC9:**	9
Additional leg created e.g. by using ICA operation. This state represents a 1-party call with a surrogate controlling leg and a joined passive leg with either an O_BCSM or T_BCSM associated with it, that is in a stable phase.	This configuration represents an O-BCSM instance associated with a Dummy Signalling Termination and connected to a peer T-BCSM. Entry events: - See exit event from SC6. Exit events: - To SC0: Release.Resp.Conf sent to a T-BCSM if sent by the initial BCSM. - To SC6: Release.Resp.Conf sent to a T-BCSM if sent by the second O-BCSM. - To SC8: Setup.Resp.Conf received from the T-BCSM associated with the second O-BCSM. A Peer T-BCSM associated with a Basic Signalling Termination. This Signalling Termination provides the interface to the outgoing Signalling Path. No signalling transparency.	

Stable_1_Party: (relay case e.g. after SplitLeg)	SC10:	10
This state represents a 1-party call with a surrogate controlling leg and a joined passive leg with either an O_BCSM or T_BCSM associated with it, that is in a stable phase. The joined passive leg that has been split (bearer connection broken) has signalling transparency with the controlling leg.	One O-BCSM instance associated with one Basic Signalling Termination that provides the interface to the incoming Signalling Path. A peer T-BCSM instance associated with one Signalling Termination that provides the interface to the outgoing Signalling Path. Signalling transparency end-to-end, i.e. relay capability for the BST. In this case the signalling connection A to B is unaffected as the bearer connection is broken. One O-BCSM instance is created and associated with a Dummy Signalling Termination for the joined passive leg.	

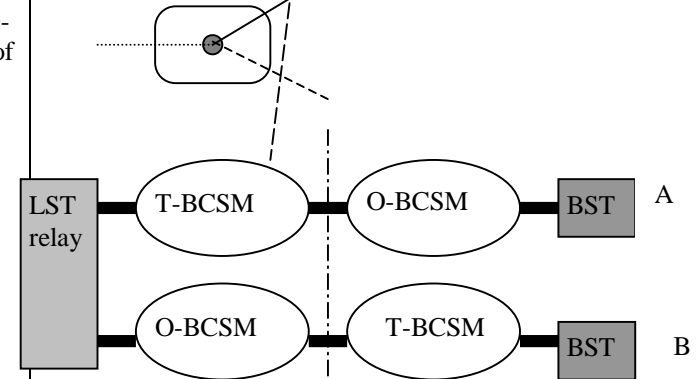| **Forward: (relay case- user initiated call)**	**SC11:**	11a
This state represents a forwarded call with a surrogate controlling leg, a joined passive leg with which an O_BCSM or a T_BCSM is associated, and a pending passive leg with which an O_BCSM is associated. Example: Call forwarding from Terminating_Setup using Connect operation; i.e. establishment of a call between a calling party A and a called party B.	This configuration represents a T-BCSM and an O-BCSM which has not left the SendCall PIC, both of which are associated with a Dummy Signalling Termination. Entry events: - See exit event from SC3. Exit events: - To SC0: Release.Resp.Conf sent by the O-BCSM to its peer T-BCSM. - To SC10: Setup.Resp.Conf received by the O-BCSM from its peer T-BCSM. **SC11a:** One O/T-BCSM instance associated with a Loop Signalling Termination. A peer O/T-BCSM instance associated with one Signalling Termination that provides the interface to the incoming/outgoing Signalling Path. This BCSM instance is not visible in the CSCV. One O-BCSM instance associated with the Loop Signalling Termination. A Peer T-BCSM is created when the O-BCSM reaches the SendCall PIC. It becomes associated with a Signalling Termination when the FacilitySelectedAndAvailable DP is reached. This Signalling Termination provides the interface to the forwarded-to Signalling Path. This BCSM is not visible in the CSCV.	

Forward: This state represents a forwarded call with a surrogate controlling leg, a joined passive leg with which an O_BCSM or a T_BCSM is associated, and a pending passive leg with which an O_BCSM is associated. Example: Call Forwarding from Stable_1-Party using Connect operation, i.e. establishment of a call between two called parties (B1 and B2). LST filter: two or more DSTs connected together.	**SC11b:** One O/T-BCSM instance associated with a Loop Signalling Termination. A peer O/T-BCSM instance associated with one Signalling Termination that provides the interface to the incoming/outgoing Signalling Path. This BCSM instance is not visible in the CSCV. One O-BCSM instance associated with the Loop Signalling Termination. A Peer T-BCSM is created when the O-BCSM reaches the SendCall PIC. It becomes associated with a Signalling Termination when the FacilitySelectedAndAvailable DP is reached. This Signalling Termination provides the interface to the forwarded-to Signalling Path. This BCSM is not visible in the CSCV.	11b
Stable_Multi-Passive_Party: (relay case – user initiated call) This state represents a transferred call with a surrogate controlling leg and two to N joined passive legs with each of which an O_BCSM or a T_BCSM is associated. The call between the involved passive legs is in the stable phase.	**SC12:** This configuration represents a T-BCSM and an O-BCSM which has left the SendCall PIC, both of which are associated with a Loop Signalling Termination. Entry events: - See exit event from SC9. Exit events: - To SC0: Release.Resp.Conf sent by the O/T-BCSM to its peer T/O-BCSM. - To SC11: Instruction resulting from a MergeCallSegments operation. - To SC1: All but one BCSM instances deleted.	12a

	SC12a: One O/T-BCSM instance associated with a Loop Signalling Termination and a peer O/T-BCSM instance associated with one Signalling Termination that provides the interface to the incoming/outgoing Signalling Path. This BCSM instance is not visible in the CSCV. One or more O-BCSM instances associated with the Loop (or Dummy) Signalling Termination each of which has a Peer T-BCSM instance associated with a Signalling Termination that provides the interface to an outgoing Signalling Path. These BCSMs are not visible in the CSCV.	
Stable Multi Passive Party: (e.g. a SCF initiated two party call) This state represents a transferred call with a surrogate controlling leg and two to N joined passive legs with each of which an O_BCSM or a T_BCSM is associated. The call between the involved passive legs is in the stable phase.	**SC12b:** This configuration represents two O-BCSM instances associated with a Loop Signalling Termination. Entry events: - See exit event from SC7. Exit events: - To SC0: Release.Resp.Conf sent by one O-BCSM to its peer T-BCSM. - To SC11: Instruction resulting from a MergeCallSegments. One O/T-BCSM instance associated with a Loop Signalling Termination and a peer O/T-BCSM instance associated with one Signalling Termination that provides the interface to the incoming/outgoing Signalling Path. This BCSM instance is not visible in the CSCV.	12b

	One or more O-BCSM instances associated with the Loop (or Dummy) Signalling Termination each of which has a Peer T-BCSM instance associated with a Signalling Termination that provides the interface to an outgoing Signalling Path. These BCSMs are not visible in the CSCV.	
Transfer: (user initiated with relay together with an SCF initiated call) Another variant is filtering on the user initiated call (e.g. follow-on call case). This state represents a transferred call with a surrogate controlling leg and two to N joined passive legs with each of which an O_BCSM or a T_BCSM is associated. The call between the involved passive legs is in the stable phase. One leg has a signalling relay capability with another leg in another call segment.	**SC12c:** One O/T-BCSM instance associated with a Basic Signalling Termination and a peer T-BCSM instance associated with one Signalling Termination that provides the interface to the incoming/outgoing Signalling Path. This BCSM instance is not visible in the CSCV. One O-BCSM instances associated with the Dummy) Signalling Termination which has a Peer T-BCSM instance associated with a Signalling Termination that provides the interface to an outgoing Signalling Path. These BCSMs are not visible in the CSCV.	12c

| Stable_Multi_Party: This state represents a multi-party call with a joined controlling leg and two to N joined passive legs with each of which an O_BCSM or a T_BCSM is associated. | **SC13:** This configuration represents two or more BCSM instances (with or without their peer BCSM instances) associated with the same Basic Signalling Termination.

 Entry events:

 - See exit event from SC10.

 Exit events:

 - To SC0: All BCSM instances deleted.

 - To SC1: All but one BCSM instances deleted.

 One or more O/T BCSM associated with the same Basic Signalling Termination. Each of which has a peer BCSM instance associated with its own Signalling Termination.

 Bearer connection exists between the three legs, but signalling transparency only between the A party and B party, not to the C-party (Dummy ST) as an additional created leg. | 13 |
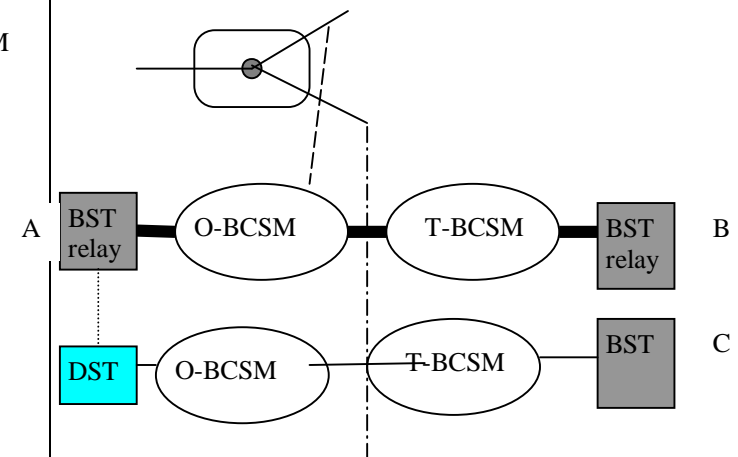
Annex D (informative):
Mapping tables for Abstract Signalling Primitives

D.1 Introduction

The signalling termination (ST) control interface is a generic interface that can be mapped to different signalling protocols.
Mapping examples are provided from basic ST_A respectively basic ST_B primitive signals to DSS1 and ISUP messages.
This is shown in mapping tables for each half call. However, mapping to other signalling protocols (e.g. H323, SIP, BICC) may also be applied.

Between the two call halves a switch-internal intra-BCSM interface (IBI) carrying abstract generic primitive signals is applied.

The primitive signals defined supporting the UNI/NNI interface is given in clause 6.
The SCF-SSF primitives supporting the INAP interface are not listed here as they map directly to the corresponding INAP operations defined in clause 11.

D.2 Examples of mapping tables for IN used primitive signals

D.2.1 How to read the tables

The mapping tables below show the generic abstract primitive signals and their interfaces and proposed possible mapping via Signalling Terminations to applicable agent signals As an example ISUP and DSS1 are here indicated as possible applied agent protocols.
However, any applicable agent protocol may be used (e.g. SIP, H323).
It may be possible to derive the actual specification for any agent protocol by using this description in combination with the appropriate interworking specification.

In the tables references are made to the primitive signalling interfaces (e.g. in forward direction the interfaces c, e and g) and to the agent signalling interfaces (e.g. a and i) as shown in figure D.1.

Originating Call Half   Terminating Call Half
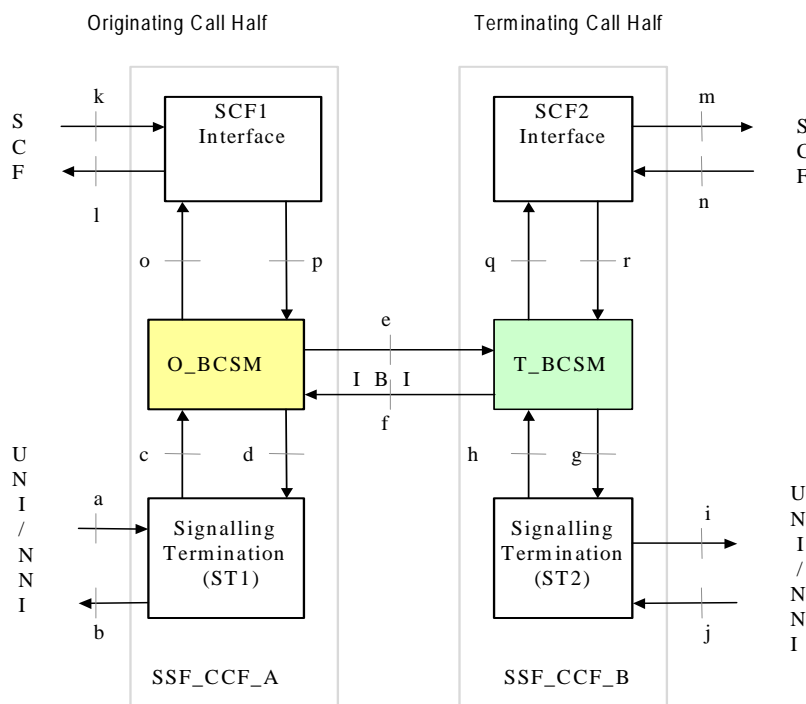
Figure D.1: Abstract Primitive Signal flow
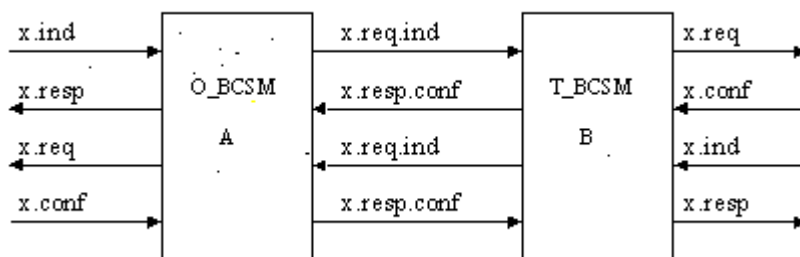
Figure D.2: Abstract Primitive signal conventions

Primitive signal types examples:

Confirmed:

EXAMPLE 1:   B party answer message received in response to a setup request messages (e.g. ANM, CON in ISUP).

Unconfirmed:

EXAMPLE 2:   B party alerted, Alert message sent backward to notify calling party (e.g. ACM (free subscriber) or CPG (Alert) in ISUP).

End-to-End:

EXAMPLE 3:   Call setup messages that need end-to-end messaging (e.g. IAM, ACM, ANM, CON in ISUP).

Link-by-Link:

EXAMPLE 4:   Release request from the network or a call party (e.g. REL/RLC in ISUP).

NOTE: The term "influence" is used in the tables to indicate where the SCF may have the ability to impact call signalling procedures, e.g. ISUP messages and parameters. Where a mapping to UNI/NNI appropriate signalling messages for IN CS-2 support is required and not known, this is indicated as to be determined (tbd).

D.2.2 Abstract Signalling Primitive signals mapping tables, originating half call

D.2.2.1 Call control Abstract Signalling Primitive Signals

Table D.1: Primitive Signals, mapping to signalling Agent Protocols, Originating Half Call

Interface Primitives	ST1 (c/d)	IBI-Signal (e/f)	Information notes	Agent Protocol ISUP (a/b)	Agent Protocol DSS.1 (a/b)
Setup	indication (c)	req.ind (e)	End-to-End	IAM	SETUP
Setup	response (d)	resp.conf (f)	End-to-End*) May be Discarded if backward resp. se sent previously.	ANM, CON *)	CONNECT *)
SubsequentAddress	indication (c)	req.ind (e)	Link-by-Link May include both address digits(s) and indication for AddressEnd	SAM	INFORMATION
CallProgress	request (d)	req.ind (f)	End-to-End *)May be discarded if sent previously	ACM, CPG	ALERTING,PROGRESS
Release	request (d)	req.ind (f)	Link-by-Link B-Party (or SSF) initiated disconnect	REL/RLC	DISCONNECT
Release	indication (c)	req.ind (e)	Link-by-Link A-Party initiated disconnect	REL/RLC	DISCONNECT
NetworkSuspend	request (d)	req.ind (f)	End-to-End CS2 *)"on-hook"	SUSPEND	- *)
NetworkResume	request (d)	req.ind (f)	End-to-End CS2 *)"off-hook"	RESUME	- *)
ServiceFeature	request (d)	req.ind (f)	'Link-by-Link' B-party initiated Midcall event -- Applies to stimulus and functional terminal protocols		
ServiceFeature	indication (c)	req.ind (e)	A-party initiated Midcall eventApplies to stimulus and functional terminal protocols	tbd	tbd
Data	request (d)	req.ind (f) req.ind (e)	ServiceToUser-Information (Sent to A, received from T_BSM or sent to B)	tbd	tbd
Data	indication (c) request (d)	req.ind (e) req.ind (f)	UserToService-Information (send by A-Party or B-Party)	tbd	tbd
Failure	indication (c)	req.ind (e) req.ind (f) *)	*) Call release is initiated by SSF	REL/RLC	DISCONNECT

D.2.2.2 SCF -SSF Abstract Signalling Primitive signals

Table D.2: Primitive Signals and mapping to SCFAgent Protocols, Originating Half Call

Interface Primitives	SCF1 (o/p)	IBI-Signal (e/f)	ST1 (c/d)	Information notes	Agent Protocols INAP (k/l)
ActivateService-Filtering	(p)	-	-	SSMESSF internal	ActivateService-Filtering
ActivityTest	(p)	-	-	*) SSF-SCF check	
ApplyCharging	(p)	-	-	-	ApplyCharging
ApplyChargingReport	(o)	-	-	-	ApplyCharging-Report
AssistRequest-Instructions	(o)	-	-		AssistRequest-Instructions
CallGap	(p)	-	-		CallGap
CallInformationReport	(o)	-	-		CallInformation-Report
CallInformationRequest	(p)	-	-		CallInformation-Request
Cancel	(p)	-	-	Cancel 'all'	Cancel
CollectInformation	(p)	-influence	influence		Collect-Information
Connect	(p)	influence	influence		Connect
ConnectToResource	(p)	-influence	-influence		ConnectTo-Resource
Continue	(p)	-	-	resumes call processing	Continue
ContinueWith-Argument	(p)	influence	influence	(INAP CS2) resumes call processing	ContinueWith-Argument
CreateCSA	(p)	-	-	(INAP CS2)	CreateCSA
DisconnectForward-Connection	(p)	-	- *)	- *) release of e.g. a temporary connection to an IP is not modelled	Disconnect-Forward-Connection
DisconnectLeg	(p)	influence	influence	(INAP CS2)	DisconnectLeg
EntityReleased	(o)	-	-	(INAP CS2)	EntityReleased
EstablishTemporary-Connection	(p)	- *)	- *)	*) set-up of a temporary connection to an IP is not modelled	Establish-Temporary-Connection
EventNotification-Charging	(o)	-	-		Event-Notification-Charging
EventReportBCSM	(o)	-	-		EventReport-BCSM
FurnishCharging-Information	(p)	-	-		FurnishCharging-Information
InitialDP	(o)	-	-		InitialDP
InitiateCallAttempt	(p)	influence	influence		InitiateCall-Attempt
ManageTriggerData	(P	-	-	(INAP CS2)	ManageTrigger-Data
MergeCallSegments	(p)	-	-	(INAP CS2)	MergeCall-Segments
MoveCallSegments	(p)	-	-	(INAP CS2)	MoveCall-Segments
MoveLeg	(p)	-	-	(INAP CS2)	Moveleg
ReleaseCall	(p)	influence	influence		ReleaseCall
ReportUTSI	(o)	influence	influence	(INAP CS2)	ReportUTSI
RequestNotification-ChargingEvent	(p)	-*)	*) influence	*)Treatment is national network specific	RequestNotification-ChargingEvent
RequestReport-BCSMEvent	(p)	-influence	-influence	E.g. request for midCall events	RequestReport-BCSMEvent
RequestReportUTSI	(p)	-	-	(INAP CS2)	RequestReport-UTSI

Interface Primitives	SCF1 (o/p)	IBI-Signal (e/f)	ST1 (c/d)	Information notes	Agent Protocols INAP (k/l)
ResetTimer	(p)	-	-		ResetTimer
SendCharging-Information			influence		
SendSTUI	(p)	influence	influence	(INAP CS2)	SendSTUI
ServiceFiltering-Response	(o)	-	-		ServiceFiltering-Response
SplitLeg	(p)	-	-		SplitLeg

D.2.3   Abstract Signalling Primitive Signals Mapping Tables, Terminating half call

D.2.3.1   Call control Abstract Signalling Primitive Signals

Table D.3: Primitive Signals, mapping to signalling termination (ST) Agent Protocols, Terminating Half Call

Interface Primitives	IBI-Signal (e/f)	ST2 (h/g)	Information notes	Agent Protocol ISUP (i/j)	Agent Protocol DSS.1 (i/j)
Setup	req.ind (e)	req (g)	End-to-End	IAM	SETUP
Setup	resp.conf (f)	conf (h)	End-to-End*) May be Discarded if backward response sent previously	ANM, CON *)	CONNECT *)
SubsequentAddress	req.ind (e)	req (g)	Link-by-Link May include both address digits(s) and indication for AddressEnd	SAM	INFORMATION
CallProgress	req.ind (f)	ind (h)	End-to-End *)May be discarded if sent previously	ACM, CPG	ALERTING, PROGRESS
Release	req,ind (f)	ind (h)	Link-by-Link B-Party initiated disconnect	REL/RLC	DISCONNECT
Release	req.ind (e)	req (g)	Link-by-Link A-Party (or SSF) initiated disconnect	REL/RLC	DISCONNECT
NetworkSuspend	req.ind (f)	ind (h)	End-to-End CS2 *)"on-hook"	SUSPEND	- *)
NetworkResume	req.ind (f)	ind (h)	End-to-End CS2 *)"off-hook"	RESUME	- *)
ServiceFeature	req.ind (f)	ind (h)	'Link-by-Link' B-party initiated Midcall event -- Applies to stimulus and functional terminal protocols	tbd	tbd
ServiceFeature	req.ind (e)	req (g)	A-party initiated Midcall eventApplies to stimulus and functional terminal protocols	tbd	tbd
Data	req.ind (f)	ind (h) req (g)	ServiceToUser-Information (Sent to A, received from T_Signal or sent to B)	tbd	tbd
Data	req.ind (e)req.ind (f)	req (g) ind (h)	UserToService-Information (Send by A-Party or B-Party)	tbd	tbd
Failure	ind.req (f) ind.req(e)	ind (h)	*) Call release is initiated by SSF	REL/RLC	DISCONNECT

D.2.3.2   SCF - SSF Abstract Signalling Primitive Signals

Table D.4: Primitive Signals and mapping to SCF Agent Protocols, Terminating half call

Interface Primitives	SCF2 (q/r)	IBI-Signal (e/f)	ST2 (h/g)	Information notes	Agent Protocols INAP (k/l)
ActivateService-Filtering	(r)	- no influence (except if call is filtered)	- no influence (except if call is filtered)	SSMESSF internal	ActivateService-Filtering
ActivityTest	(r)	-	-	*) SSF-SCF dialogue check	
ApplyCharging	(r)	-	-	-	ApplyCharging
ApplyCharging-Report	(q)	-	-	-	ApplyCharging-Report
AssistRequest-Instructions	(q)	-	-		AssistRequest-Instructions
CallGap	(r)	- no influence (except if call is gapped)	- no influence (except if call is gapped)		CallGap
CallInformation-Report	(q)	-	-		CallInformation-Report
CallInformation-Request	(r)	-	-		CallInformation-Request
Cancel	(r)	-	-	Cancel 'all'	Cancel
Connect	(r)	influence	influence		Connect
ConnectToResource	(r)	-influence	-influence		ConnectTo-Resource
Continue	(r)	-	-		Continue
ContinueWith-Argument	(r)	influence	influence	(INAP CS2)	ContinueWith-Argument
CreateCSA	(r)	-	-	(INAP CS2)	CreateCSA
DisconnectForward-Connection	(r)	-	- *)	*) release of e.g. a temporary connection to an IP is not modelled	Disconnect-Forward-Connection
DisconnectLeg	(r)	influence	influence	(INAP CS2)	DisconnectLeg
EntityReleased	(q)	-	-	(INAP CS2)	EntityReleased
EstablishTemporary-Connection	(r)	- *)	- *)	*) set-up of a temporary connection to an IP is not modelled	Establish-Temporary-Connection
EventNotification-Charging	(q)	-	-		Event-Notification-Charging
EventReportBCSM	(q)	-	-		EventReport-BCSM
FurnishCharging-Information	(r)	-	-		FurnishCharging-Information
InitialDP	(q)	-	-		InitialDP
InitiateCallAttempt	(r)	influence	influence		InitiateCall-Attempt
ManageTriggerData	(r)	-	-	(INAP CS2)	ManageTrigger-Data
MergeCallSegments	(r)	-	-	(INAP CS2)	MergeCall-Segments
MoveCallSegments	(r)	-	-	(INAP CS2)	MoveCall-Segments
MoveLeg	(r)	-	-	(INAP CS2)	Moveleg
ReleaseCall	(r)	influence	influence		ReleaseCall
ReportUTSI	(q)	influence	influence	(INAP CS2)	ReportUTSI
RequestNotification-ChargingEvent	(r)	influence *)	-*)	*)Treatment is national network specific	Request-Notification-ChargingEvent

Interface Primitives	SCF2 (q/r)	IBI-Signal (e/f)	ST2 (h/g)	Information notes	Agent Protocols INAP (k/l)
RequestReport-BCSMEvent	(r)	-influence	-influence	E.g. request for midCall events	RequestReport-BCSMEvent
RequestReportUTSI	(r)	-	-	(INAP CS2)	RequestReport-UTSI
ResetTimer	(r)	-	-		ResetTimer
SendCharging-Information	(r)	influence	-		SendCharging-Information
SendSTUI	(r)	influence	influence	(INAP CS2)	SendSTUI
ServiceFiltering-Response	(q)	-	-		ServiceFiltering-Response
SplitLeg	(r)	-	-	(INAP CS2)	SplitLeg

Annex E (informative): Global Call Reference

E.1 Introduction

The purpose of the new ISUP Global Call Reference parameter is to extend the identification of a call on a global level. A tentative coding of this parameter has been included in ISUPv4.

The provisional coding for the Global Call Reference parameter as attached. This coding consists of the combination of a network identity field (identical to the Network Identification in ES 201 296) and a call reference field (variable length to cover the extended CIC length in BICC (4octet CIC)) and possible message in an IP environment.

Changes to ITU-T Recommendation Q.762.

3.z **Global Call Reference(GCR):** Information sent in the forward direction to correlate call activities.

Changes to ITU-T Recommendation Q.763.

The following is the proposed coding of this parameter.

E.1.1 Global Call Reference

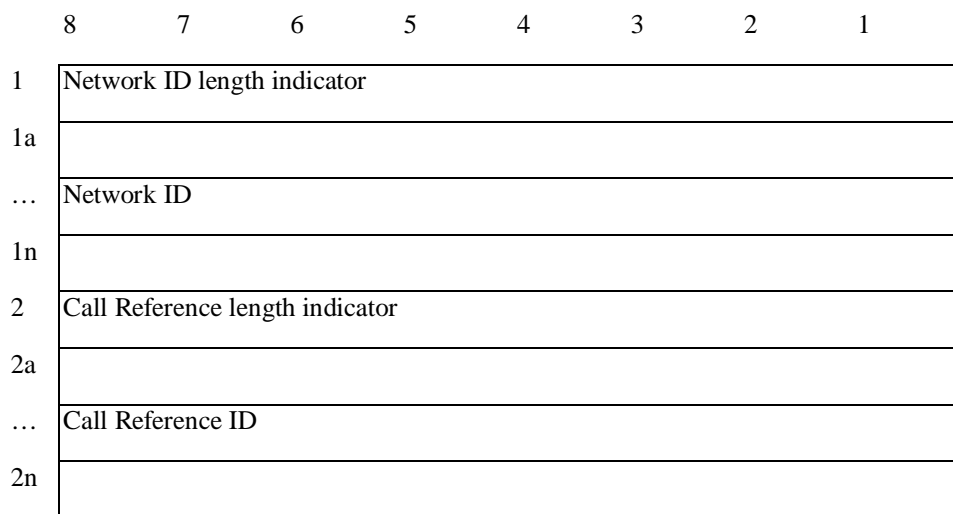|	8	7	6	5	4	3	2	1
1	Network ID length indicator							
1a								
…	Network ID							
1n								
2	Call Reference length indicator							
2a								
…	Call Reference ID							
2n								

Figure E.1: Global Call Reference

The following codes are used in the subfields of the network identity parameter field:

1) Network ID

   The following information is used in this subfield of the Global Call Reference parameter field using ASN.1.

   NetworkIdentification::= OBJECT IDENTIFIER.

-- Following structure of the networkIdentification value shall be used:

-- {itu-t (0) administration (2) national regulatory authority (x) network (y) node identification (z)}.

-- The value for x is the value of the national regulatory authority, the value for y is under the control.

-- of the national regulatory authority concerned, the value for z is under the control of the network concerned.

-- The data Country code specified in ITU-T Recommendation X.121 shall be used for "national regulatory".

-- "authority".

The Basic Encoding Rules (BER) according to ITU-T Recommendation X.690 shall be used

2) Call Reference

   A binary number used for the call reference of the call. This is generated by the node for each call.

History

Document history				
V1.1.1	August 2000	Public Enquiry	PE 20001215:	2000-08-16 to 2000-12-15
V1.1.2	July 2001	Vote	V 20010831:	2001-07-02 to 2001-08-31
V1.1.2	September 2001	Publication		