EUROPEAN TELECOMMUNICATION STANDARD

ETS 300 374-1

September 1994

Source: ETSI TC-SPS

Reference: DE/SPS-03015

ICS: 33.020, 33.080

Key words: IN, CS1, INAP

Intelligent Network (IN); Intelligent Network Capability Set 1 (CS1); Core Intelligent Network Application Protocol (INAP); Part 1: Protocol specification

ETSI

European Telecommunications Standards Institute

ETSI Secretariat

Postal address: F-06921 Sophia Antipolis CEDEX - FRANCE
Office address: 650 Route des Lucioles - Sophia Antipolis - Valbonne - FRANCE
X.400: c=fr, a=atlas, p=etsi, s=secretariat - **Internet:** secretariat@etsi.fr

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

Contents

Foreword

This European Telecommunication Standard (ETS) has been produced by the Signalling Protocols and Switching (SPS) Technical Committee of the European Telecommunications Standards Institute (ETSI).

This ETS is based on ITU-T Recommendation Q.1218 (1993). It provides major modifications and further requirements to this base document.

This ETS is part 1 of a multi-part standard covering the Capability Set 1 (CS1) core Intelligent Network Protocol (INAP) as described below:

Part 1: "Protocol specification";

Part 2: "Protocol Implementation Conformance Statement (PICS) proforma";

Part 3: "Test Suite Structure and Test Purposes (TSS&TP)";

Part 4: "Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma";

Part 5: "Protocol specification for the SCF-SDF interface";

Part 6: "PICS proforma for the SCF-SDF interface".

Transposition dates	
Date of latest announcement of this ETS (doa):	31 December 1994
Date of latest publication of new National Standard or endorsement of this ETS (dop/e):	30 June 1995
Date of withdrawal of any conflicting National Standard (dow):	30 June 1995

Blank page

1 Scope

This first part of ETS 300 374 defines the Intelligent Network Application Protocol (INAP) required for support of Capability Set 1 (CS1). It supports interactions between the following three Functional Entities (FEs), as defined in the Intelligent Network (IN) functional model:

- Service Switching Function (SSF);

- Service Control Function (SCF);

- Specialized Resource Function (SRF).

The scope of this ETS is the further development of the INAP for both the Integrated Services Digital Network (ISDN) and Public Switched Telecommunications Network (PSTN).

It is intended as a guide to implementors and network operators to ensure interworking between different manufacturers equipment for the following IN CS1 defined interfaces (SCF-SSF and SCF-SRF).

As this ETS is intended for the early introduction of IN in the existing ISDN/PSTN, only simple solutions are assumed for solving the service interaction problems between IN and ISDN/PSTN.

NOTE: More sophisticated solutions for the service interactions between IN and the ISDN/PSTN environment should be studied in the scope of future versions of INAP and the ISDN/PSTN signalling standards.

2 Normative references

This ETS incorporates by dated and undated reference, provisions from other publications. These normative references are cited at the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to this ETS only when incorporated in it by amendment or revision. For undated references the latest edition of the publication referred to applies.

[1] ETS 300 008 (1993): "Integrated Services Digital Network (ISDN); CCITT Signalling System No.7; Message Transfer Part (MTP) to support international interconnection".

[2] ETS 300 009 (1994): "Integrated Services Digital Network (ISDN); CCITT Signalling System No.7; Signalling Connection Control Part (SCCP) [connectionless and connection-oriented] to support international interconnection".

[3] ETS 300 121 (1992): "Integrated Services Digital Network (ISDN); Application of the ISDN User Part (ISUP) of CCITT Signalling System No.7 for international ISDN interconnections (ISUP version 1)".

[4] ETS 300 196-1 (1993): "Integrated Services Digital Network (ISDN); Generic functional protocol for the support of supplementary services; Digital Subscriber Signalling System No. one (DSS1) protocol; Part 1: Protocol specification".

NOTE: ETS 300 196-1 (1993) was initially published as ETS 300 196 (1993).

[5] ETS 300 287 (1993): "Integrated Services Digital Network (ISDN); CCITT Signalling System No.7; Transaction Capabilities Application Part (TCAP) version 2".

[6] ETS 300 348 (1994): "Intelligent Network (IN); Physical plane for intelligent network Capability Set 1 (CS1) [ITU-T Recommendation Q.1215 (1993)]".

[7] ETS 300 356-1: "Integrated Services Digital Network (ISDN); CCITT Signalling System No.7; ISDN User Part (ISUP) version 2 for the international interface; Part 1: Basic services".

[8] ETS 300 403-1: "Integrated Services Digital Network (ISDN); Digital Subscriber Signalling System No. one (DSS1); User-network interface layer 3 specification for basic call control; Part 1: Protocol specification [ITU-T Recommendation Q.931 (1993), modified]".

[9] ITU-T Recommendation Q.700 (1993): "Introduction to CCITT Signalling System No.7".

[10] ITU-T Recommendation Q.773 (1993): "Specifications of Signalling System No.7; Transaction Capabilities formats and encoding".

[11] ITU-T Recommendation Q.1214 (1993): "Distributed functional plane for intelligent network CS1".

[12] ITU-T Recommendation Q.1218 (1993): "Interface Recommendation for intelligent network CS1".

[13] ITU-T Recommendation Q.1400 (1993): "Architecture framework for the development of signalling and organization, administration and maintenance protocols using OSI principles".

[14] CCITT Recommendation X.208 (1988): "Specification of Abstract Syntax Notation One (ASN.1)".

[15] CCITT Recommendation X.209 (1988): "Specification of basic encoding rules for Abstract Syntax Notation One (ASN.1)".

[16] CCITT Recommendation X.219 (1988): "Remote operations: Model, notation and service definition".

[17] CCITT Recommendation X.229 (1988): "Remote operations: Protocol specification".

[18] ISO 9545 (1989): "Information technology - Open Systems Interconnection - Application Layer structure".

3 Definitions and abbreviations

For the purposes of this ETS, the following abbreviations apply:

AC	Application Context
AE	Application Entity
AEI	Application Entity Invocation
ASE	Application Service Element
APDU	Application Protocol Data Unit
ASN.1	Abstract Syntax Notation One
BCSM	Basic Call State Model
CCF	Call Control Function
CS1	Capability Set 1

DP	Detection Point
DSS1	Digital Subscriber Signalling System No. One
EDP	Event Detection Point
EDP-N	Event Detection Point - Notification
EDP-R	Event Detection Point - Request
FCI	Furnish Charging Information
FE	Functional Entity
FEAM	Functional Entity Access Manager
FSM	Finite State Model
ID	Identifier
IN	Intelligent Network
INAP	Intelligent Network Application Protocol
IP	Intelligent Peripheral
ISDN	Integrated Services Digital Network
ISPBX	Integrated Services Private Branch eXchange
ISUP	ISDN User Part
LE	Local Exchange
MACF	Multiple Association Control Function
MTP	Message Transfer Part
P&C	Prompt and Collect
PA	Play Announcement
PDU	Protocol Data Unit
PE	Physical Entity
PSTN	Public Switched Telecommunication Network
RCO	Resource Control Object
ROSE	Remote Operations Service Element
SACF	Single Association Control Function
SAO	Single Association Object
SCCP	Signalling Connection Control Part
SCF	Service Control Function
SCME	SCF Management Entity
SCP	Service Control Point
SCSM	SCF Call State Model
SDF	Service Data Function
SDP	Service Data Point
SL	Service Logic
SLP	Service Logic Program
SLPI	Service Logic Program Instance
SMF	Service Management Function
SRF	Specialized Resource Function
SRME	SRF Management Entity
SRSM	SRF Call State Model
SSF	Service Switching Function
SSME	SSF Management Entity
SSN	Sub-System Number
SSP	Service Switching Point
TC	Transaction Capabilities
TCAP	Transaction Capabilities Application Part
TDP	Trigger Detection Point
TDP-N	Trigger Detection Point - Notification
TDP-R	Trigger Detection Point - Request

4 General

4.1 Definition methodology

The definition of the protocol is split into three Clauses:

- the definition of the Single/Multiple Association Control Function (SACF/MACF) rules for the protocol (Clause 5);

- the definition of the operations transferred between entities (Clause 6);

- the definition of the actions taken at each entity (Clause 7).

The SACF/MACF rules are defined in prose. The operation definitions are in Abstract Syntax Notation 1 (ASN.1, see CCITT Recommendation X.208 [14]), and the actions are defined in terms of state transition diagrams. Further guidance on the actions to be performed on receipt of an operation can be gained from Clause 6 and from the relevant detailed procedures in Clause 7.

The INAP is a Remote Operations Service Element (ROSE) user protocol (see CCITT Recommendations X.219 [16] and X.229 [17]). The ROSE protocol is contained within the Digital Subscriber Signalling System No. One (DSS1, see ETS 300 196-1 [4]) and the Component Sublayer of the Transaction Capabilities Application Part (TCAP, see ETS 300 287 [5]).

NOTE 1: At present, the ROSE Application Protocol Data Units (APDUs) are conveyed in Transaction sublayer messages in Signalling System No.7 (Signalling System No.7) and in the Q.931 FACILITY and Call Control messages in DSS1 (see ETS 300 403-1 [8]). Other supporting protocols may be added at a later date.

NOTE 2: The INAP (as a ROSE user) and the ROSE protocol have been specified using ASN.1. At present, the only standardized way to encode the resulting PDUs is the Basic Encoding Rules (see CCITT Recommendation X.209 [15]).

4.2 Example physical scenarios

The protocol supports any mapping of functional to Physical Entities (PEs), see ETS 300 348 [6] (ITU-T Recommendation Q.1215). It is the responsibility of network operators and equipment manufacturers to decide how to co-locate FEs to the best possible advantage as this may vary between manufacturers and between network operators. Therefore the protocol is defined assuming maximum distribution (i.e. one PE per FE).

The figures depicted in this subclause show how INAP would be supported in an Signalling System No.7 network environment. This does not imply that only Signalling System No.7 may be used as the network protocol to support INAP.

When TCAP appears in one of the following figures, it is to be understood as representing the TCAP functionalities associated with a single dialogue and transaction (as opposed to a TCAP entity).

For each physical scenario, the typical SRF control procedures to be applied are shown in subclause 7.3.5.

The interface between remotely located SCF and SDF is INAP using TCAP which in turn uses the services of the connectionless SCCP and MTP (see figure 1). The SDF is responsible for any interworking to other protocols to access other types of networks.

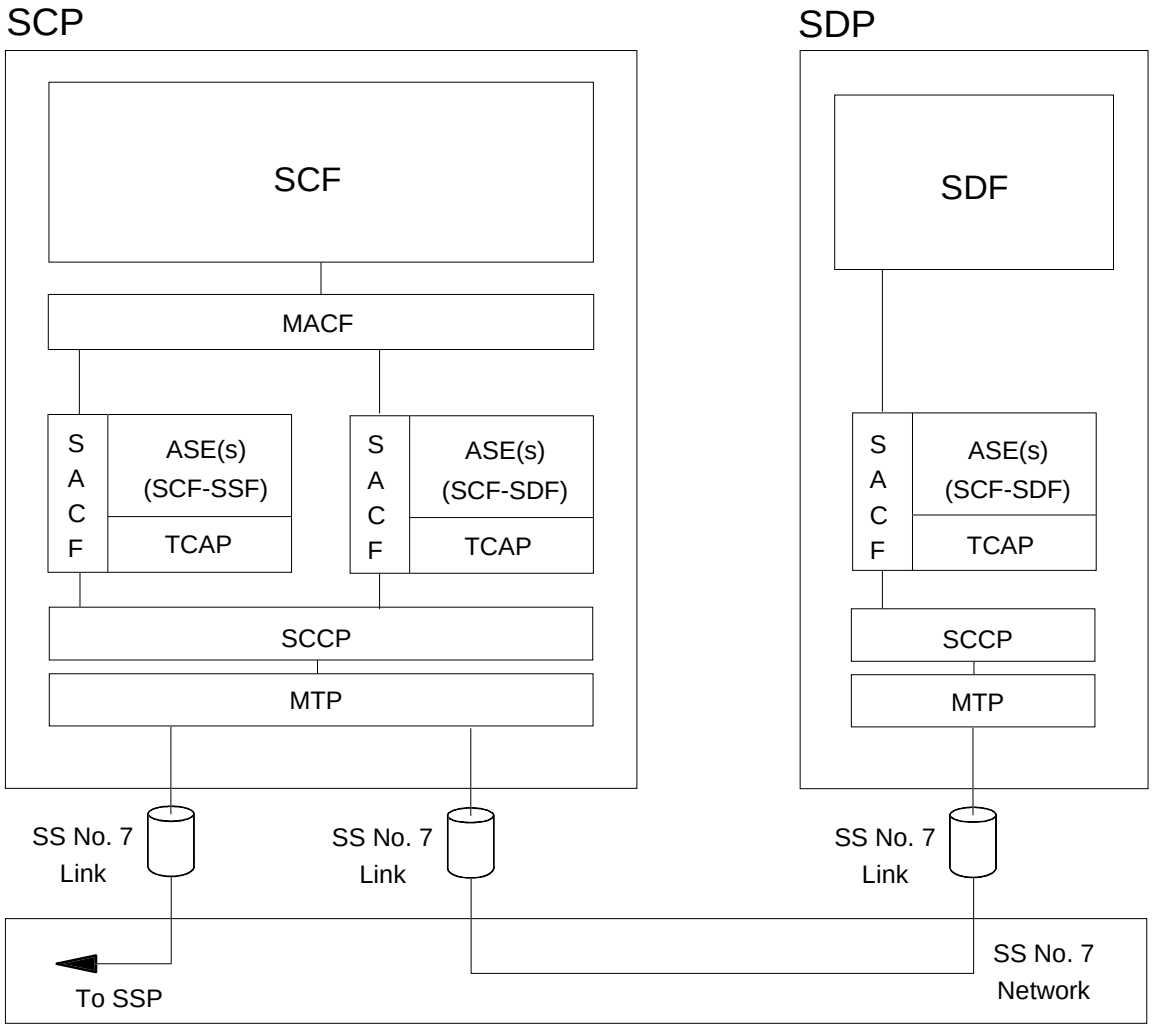


Figure 1: Physical interface between SCP and SDP

A number of example scenarios have been identified for support of the SCF, SSF and SRF FEs as PEs. These are illustrated as figures 2 to 6. Each example is characterized by:

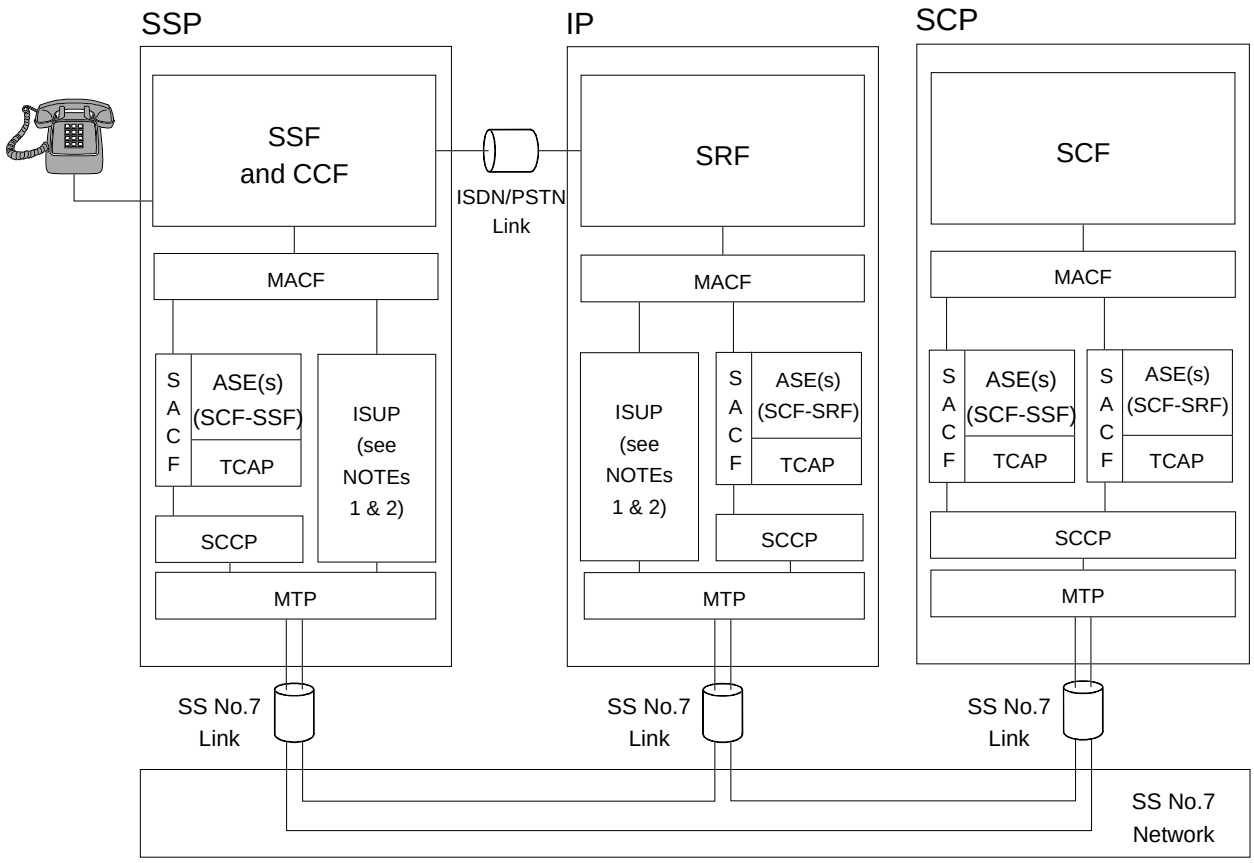
a) the method to support SCF-SRF relationship; and

b) the type of signalling system between SSF and SRF.

Table 1 summarizes the selection of features for each figure.

Table 1

Type of signalling system	Method to support SCF-SRF relationship	
between SSF and SRF	direct TCAP link	relay via SSP
ISUP	figure 2	figure 5
DSS1	figure 6	figure 3
implementation dependent	as figures 2 or 6 but with implementation dependent SCP-IP interface	figure 4

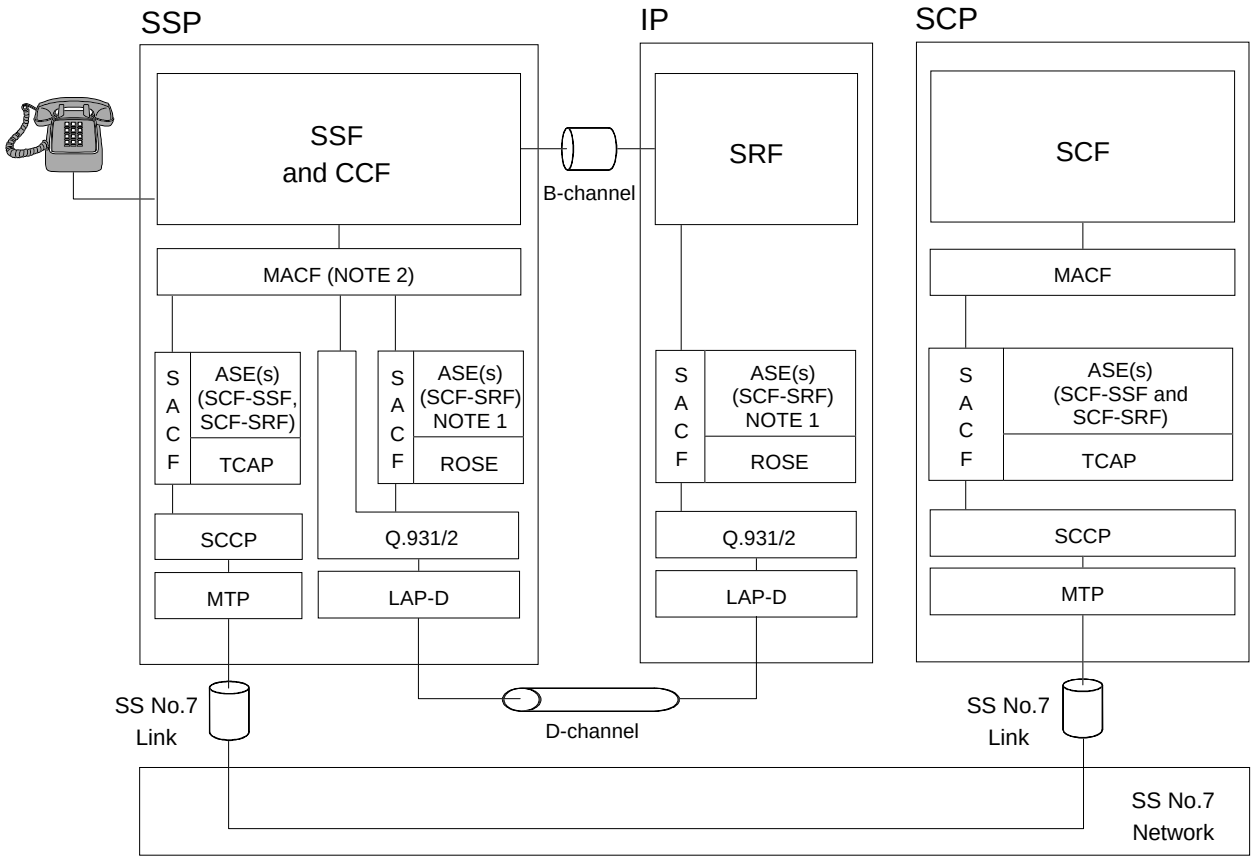


NOTE 1: Transfer of correlation information needs to be supported. This may be supported in ISUP without introducing new ISUP parameters.

NOTE 2: Other signalling systems may be used.

NOTE 3: All associations are supported by Signalling System No.7, either TCAP or ISUP. In this case, the IP is one of the network nodes.

Figure 2: Example architecture for supporting SRF, case 1 (SRF in IP connected to SSP and accessed by SCP through direct Signalling System No.7 connection)

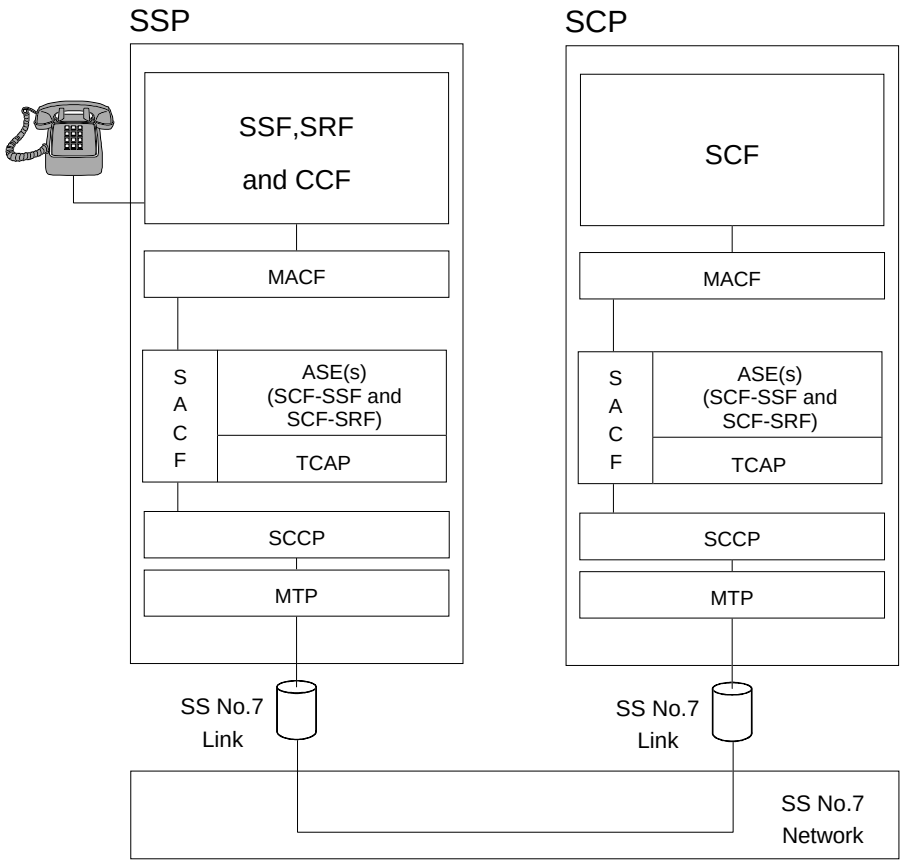


NOTE 1: Information flows between SCF and SRF are supported by this (ROSE) entity.

NOTE 2: Relay function is provided either by MACF or by application process at SSP.

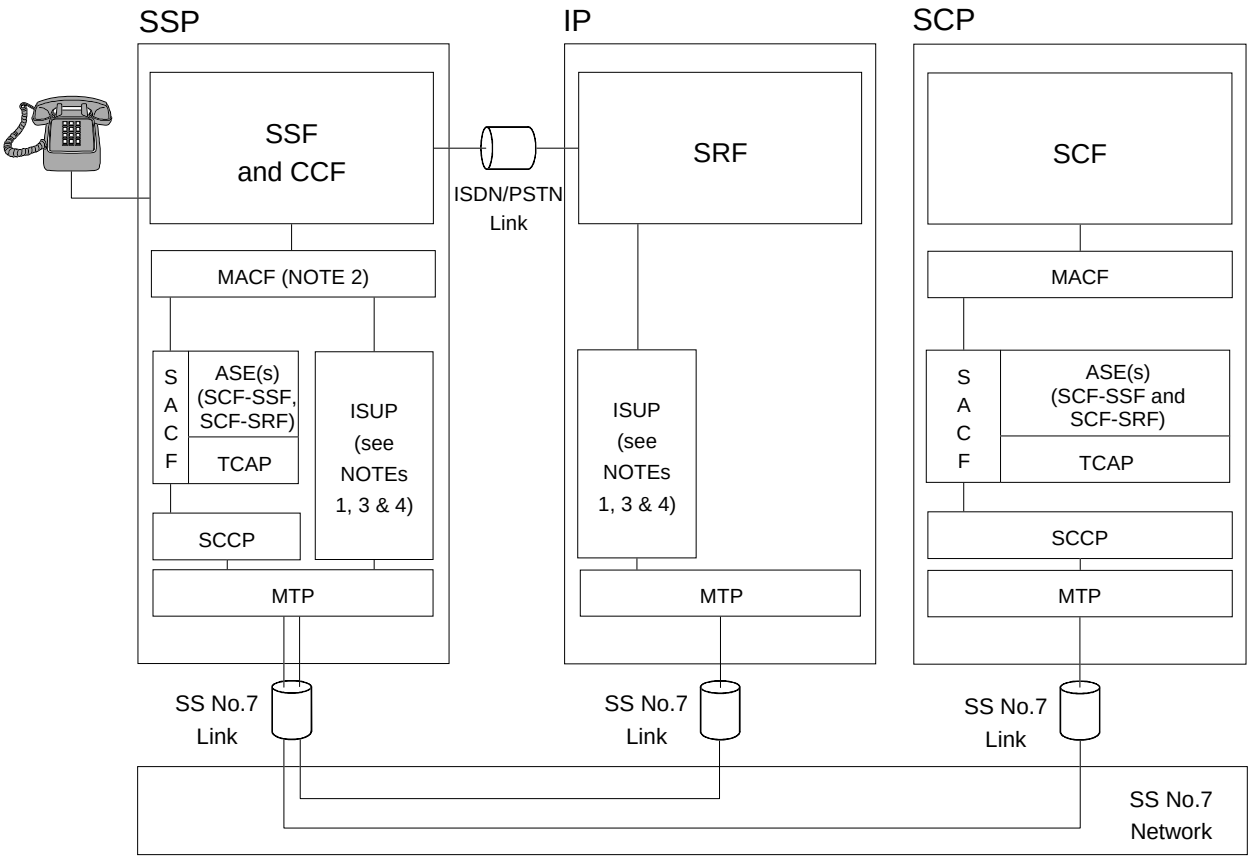
NOTE 3: IP can be accessed by DSS1 only. The IP can be a PE residing outside the network.

Figure 3: Example architecture for supporting SRF, case 2
(SRF in IP connected to SSP and accessed by SCP through D-channel via SSP)



NOTE: SSP supports both Call Control Function/Service Switching Function (CCF/SSF) and SRF. The handling of SRF by SCF could be the same as that of figure 3.

Figure 4: Example architecture for supporting SRF, case 3 (SRF in SSP and accessed via application process of SSP)



NOTE 1: Information flows between SCF and SRF as well as connection control are indirectly supported by ISUP.

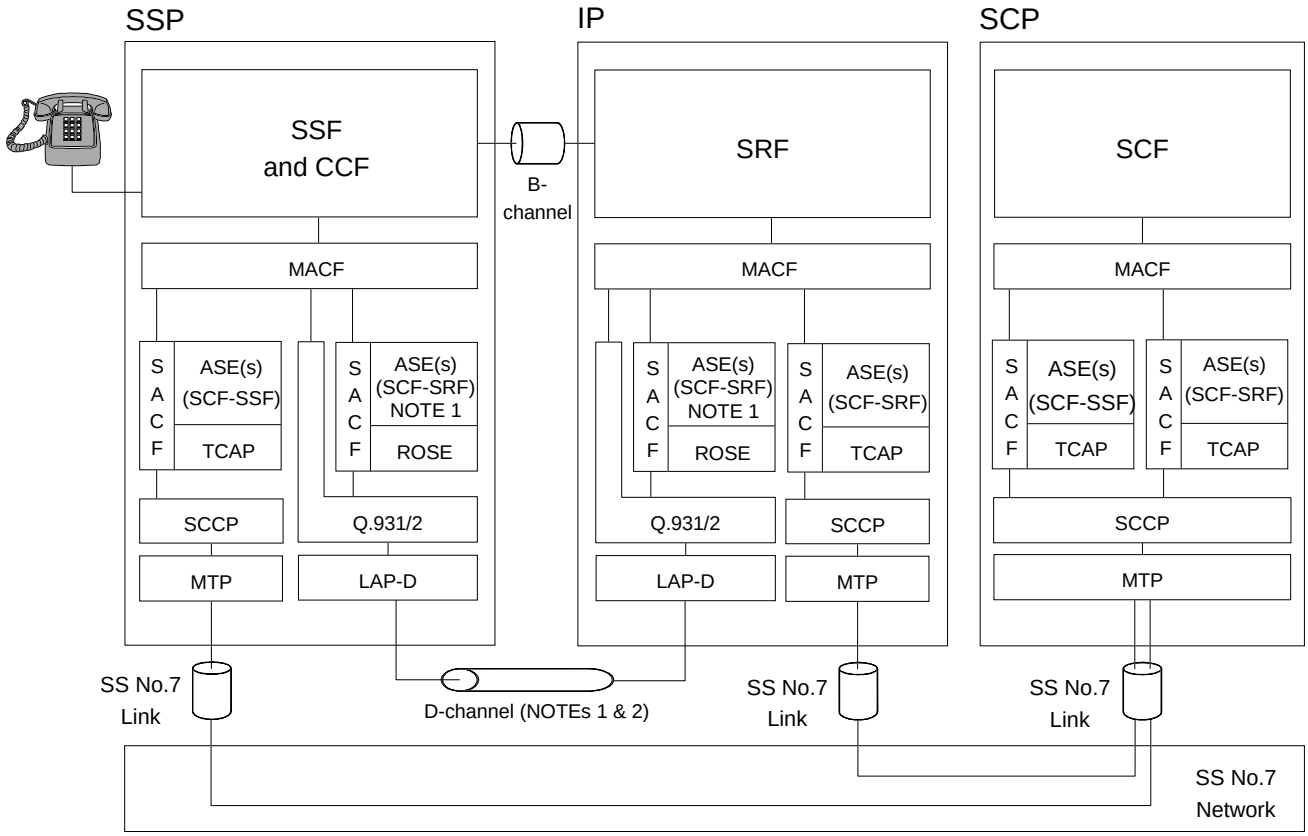
NOTE 2: Relay function is provided either by MACF or by application process at SSP.

NOTE 3: Assumes that ISUP provides a means to transport ROSE information.

NOTE 4: Other signalling systems may be used.

NOTE 5: IP can be accessed by ISUP only. The handling of SRF by SCF could be the same as that of figure 3.

Figure 5: Example architecture for supporting SRF, case 4
(SRF in IP connected to SSP and accessed by SCP through ISUP via SSP)



NOTE 1: Transfer of correlation information needs to be supported.

NOTE 2: Other signalling systems may be used.

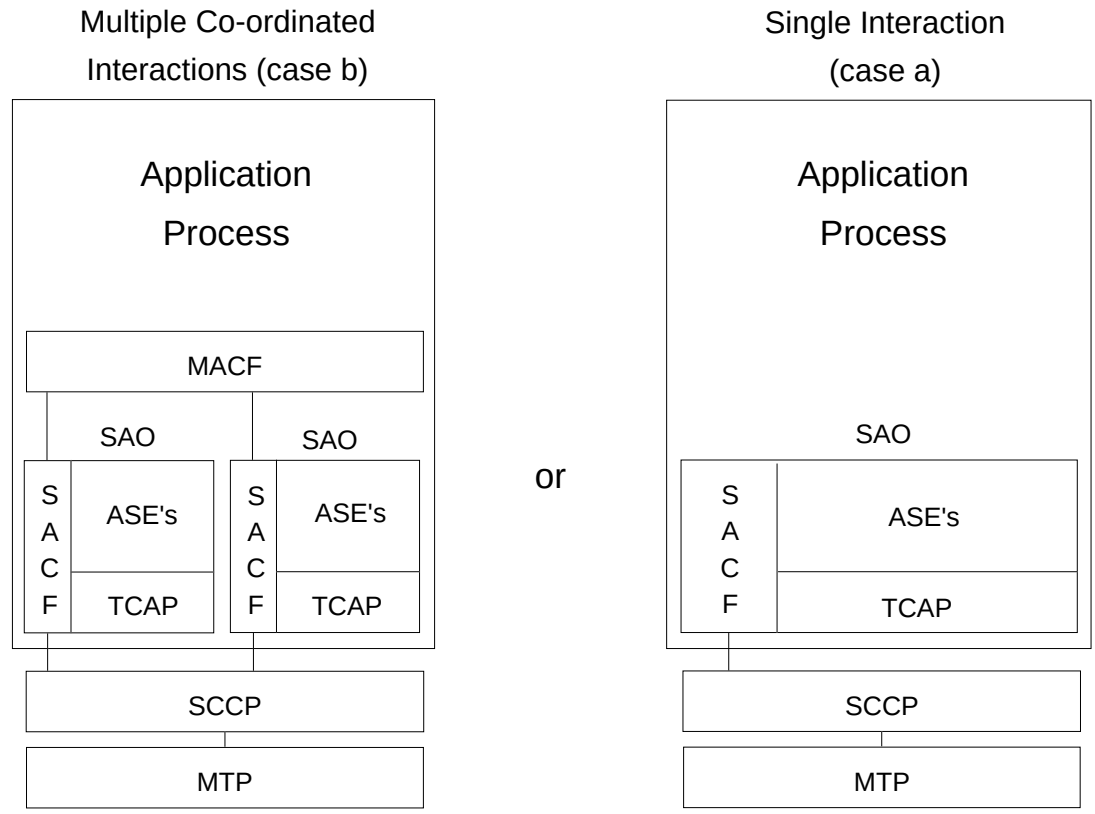
NOTE 3: The handling of SRF by SCF could be the same as that of figure 2. Other types of signalling systems could be used.

Figure 6: Example Architecture for supporting SRF, case 5 (SRF in IP connected to SCP and SSP and accessed via both Signalling System No.7 link and D-channel, respectively)

4.3 INAP protocol architecture

Many of the terms used in this subclause are based on the OSI Application Layer Structure as defined in ISO 9545 [18].

The INAP protocol architecture can be illustrated as shown in figure 7.



NOTE: INAP is the collection of specifications of all IN ASEs.

Figure 7: INAP protocol architecture

A PE has either single interactions (case a) or multiple co-ordinated interactions (case b) with other PEs.

In case a, SACF provides a co-ordination function in using Application Service Elements (ASEs), which includes the ordering of operations supported by ASE(s), (based on the order of received primitives). The Single Association Object (SAO) represents the SACF plus a set of ASEs to be used over a single interaction between a pair of PEs.

In case b, MACF provides a co-ordinating function among several SAOs, each of which interacts with an SAO in a remote PE.

Each ASE supports one or more operations. Description of each operation is tied with the action of corresponding FE modelling (see ITU-T Recommendation Q.1214 [11] and Clause 7 of this ETS). Each operation is specified using the operation macro described in figure 8.

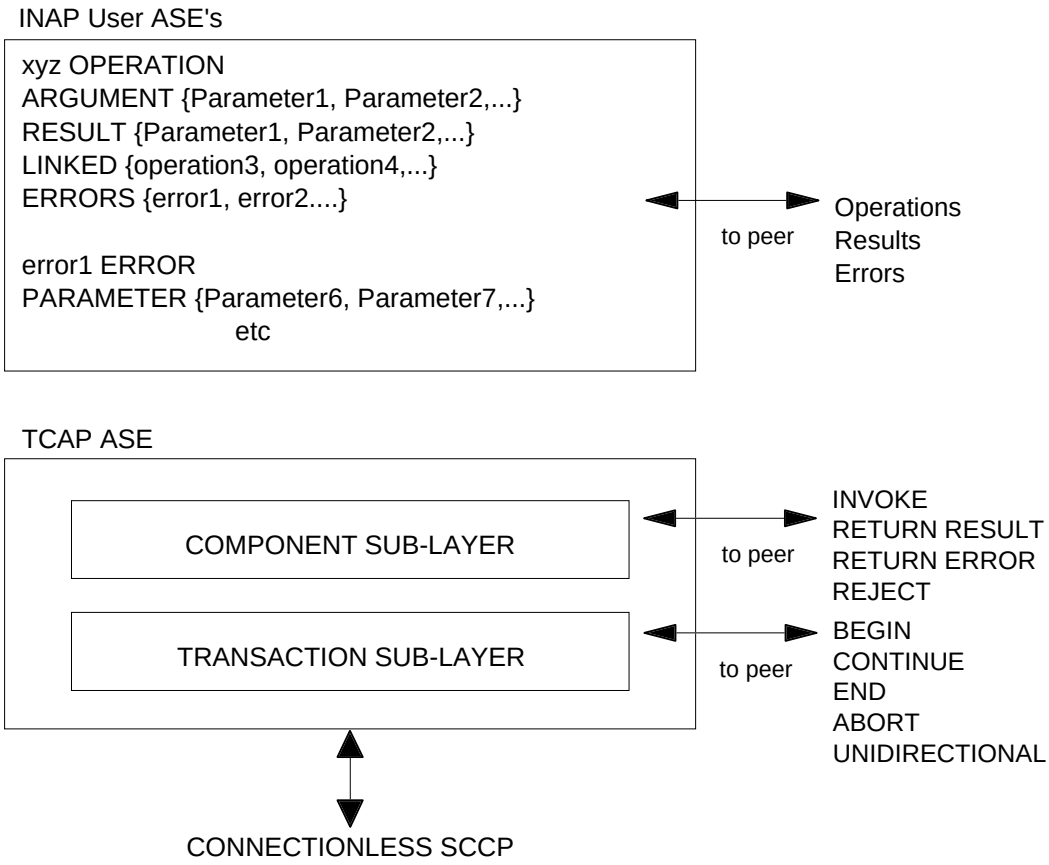


Figure 8: Operation description

The use of the Application Context (AC) negotiation mechanism (as defined in ETS 300 287 [5]) allows the two communicating entities to identify exactly what their capabilities are and also what the capabilities required on the interface should be. This should be used to allow evolution through capability sets.

If the indication of a specific AC is not supported by a pair of communicating FEs, some mechanism to pre-arrange the context shall be supported.

4.3.1 INAP signalling congestion control for Signalling System No.7

The same type of procedure shall apply as defined for ISDN User Part signalling congestion control. The INAP procedures for signalling congestion control shall as far as possible be aligned with the ISDN User Part signalling congestion control procedures as specified in ETS 300 121 [3] (CCITT Recommendation Q.767, § D.2.11), i.e. on receipt of N-PCSTATE indication primitive with the information "signalling point congested" from SCCP, the INAP shall reduce the traffic load (e.g. InitialDP and InitiateCallAttempt) into the affected direction in several steps.

The above procedure may only apply to traffic which uses MTP Point Code addressing in the affected direction.

4.4 INAP addressing

SCCP Global Title (see ETS 300 009 [2]) and MTP Point Code addressing (see ETS 300 008 [1]) ensure that PDUs reach their physical destination (i.e. the correct point code) regardless of which network it is in.

Within a node, it is the choice of the network operator/implementor as to which Sub-System Number(s) (SSN(s)) are assigned to INAP.

Regardless of the above, any addressing scheme supported by the SCCP may be used.

4.5 Relationship between ITU-T Recommendation Q.1214 and this ETS

Table 2 gives a complete list of information flows. These map one to one with operations except where indicated.

Table 2

Q.1214 reference	Information flow	Operation
6.4.2.1	ActivateServiceFiltering	Same
6.4.2.2	ActivityTest	Same
6.4.2.3	ActivityTestResponse	return result from ActivityTest
6.4.2.4	AnalyzedInformation	InitialDP, EventReportBCSM
6.4.2.5	AnalyzeInformation	Connect
6.4.2.6	ApplyCharging	Same
6.4.2.7	ApplyChargingReport	Same
6.4.2.8	AssistRequestInstructions	Same
6.4.2.9	CallGap	Same
6.4.2.10	CallInformationReport	Same
6.4.2.11	CallInformationRequest	Same
6.4.2.14	CollectedInformation	InitialDP, EventReportBCSM
6.4.2.15	CollectInformation	Same
6.4.2.16	Connect	Same
6.4.2.17	ConnectToResource	Same
6.4.2.18	Continue	Same
6.4.3.19	DisconnectForwardConnection	Same
6.4.2.20	EstablishTemporaryConnection	Same
6.4.2.21	EventNotificationCharging	Same
6.4.2.22	EventReportBCSM	Same
6.4.2.23	FurnishChargingInformation	Same
6.4.2.24	Holdcallinnetwork	ResetTimer and FurnishChargingInformation
6.4.2.25	InitialDP	Same
6.4.2.26	InitiateCallAttempt	Same
6.4.2.27	OAnswer	InitialDP, EventReportBCSM
6.4.2.28	OCalledPartyBusy	InitialDP, EventReportBCSM
6.4.2.29	ODisconnect	InitialDP, EventReportBCSM
6.4.2.30	O_MidCall	InitialDP, EventReportBCSM
6.4.2.31	O_No_Answer	InitialDP, EventReportBCSM
6.4.2.32	OriginationAttemptAuthorized	InitialDP
6.4.2.33	ReleaseCall	Same
6.4.2.34	RequestNotificationChargingEvent	Same
6.4.2.35	RequestReportBCSMEvent	Same
6.4.2.37	ResetTimer	Same
6.4.2.38	RouteSelectFailure	InitialDP, EventReportBCSM
6.4.2.40	SelectRoute	Connect
6.4.2.41	SendChargingInformation	Same
6.4.2.42	ServiceFilteringResponse	Same
6.4.2.44	TAnswer	InitialDP, EventReportBCSM
6.4.2.45	TCalledPartyBusy	InitialDP, EventReportBCSM
6.4.2.46	TDisconnect	InitialDP, EventReportBCSM
6.4.2.47	TermAttemptAuthorized	InitialDP
6.4.2.48	T_MidCall	InitialDP, EventReportBCSM
6.4.2.49	TNoAnswer	InitialDP, EventReportBCSM
6.5.2.1	AssistRequestInstructionsfromSRF	AssistRequestInstructions
6.5.2.2	CancelAnnouncement	Cancel
6.5.2.3	CollectedUserInformation	return result from PromptAndCollectUserInformation
6.5.2.4	PlayAnnouncement	Same
6.5.2.5	PromptAndCollectUserInformation	Same
6.5.2.6	SpecializedResourceReport	Same

4.6 Compatibility mechanisms used for INAP

4.6.1 Introduction

This subclause specifies the compatibility mechanisms that shall be used for INAP, in this subclause referred to as ETSI INAP to avoid confusion with INAP according to ITU-T Recommendation Q.1218 [12].

Two major categories of compatibility are handled by these mechanisms:

- compatibility with the ITU-T Recommendation Q.1218 [12] version of CS1 INAP;

- compatibility with future versions of ETSI INAP.

The second category has three sub-categories of compatibility dealt with in this subclause:

- minor changes to the ETSI INAP in future standardized versions:

  a minor change can be defined as a change of a functionality which is not essential for the requested IN service. In case it is a modification of an existing function, it is acceptable that the addressed function is executed in either the older or the modified variant. If the change is purely additional, it is acceptable that it is not executed at all and that the peer Application Entity (AE) need not know about the effects of the change. For minor changes, a new AC is not required;

- major changes to the ETSI INAP in future standardized versions:

  a major change can be defined as a change of a functionality which is essential for the requested IN service. In case it is a modification of an existing function, both application entities shall have a shared knowledge about the addressed functional variant. If the change is purely additional, the requested IN service will not be provided if one of the application entities does not support the additional functionality. For major changes, a new AC is required;

- network specific changes to ETSI INAP:

  these additions may be of either the major or minor type for a service. No new AC is expected to be defined for this type of change. At the time of definition, the additions would not be expected to be included in identical form in future versions of the ETS.

4.6.2 Definition of ETSI INAP compatibility mechanisms

4.6.2.1 Compatibility mechanism for interworking of ETSI INAP and ITU-T Q.1218 INAP

On receipt of an operation according to ITU-T Recommendation Q.1218 [12] which is not part of the ETSI INAP or is part of the ETSI INAP but which contains parameters which are not part of the ETSI INAP:

- the SSF shall apply the normal error handling for unknown operations or parameters, i.e. the normal error handling procedures as specified in Clause 10 shall be followed;

- the SCF shall apply the normal error handling for unknown operations or parameters except for parameters in the InitialDP operation. All parameters specified in ITU-T Recommendation Q.1218 [12] for InitialDP shall be known by the SCF, those not included in the ETSI INAP shall be ignored.

Additionally, tagging of ETSI INAP additions to ITU-T Recommendation Q.1218 [12] are specified from 30 downwards to avoid overlap with future additions in ITU-T Recommendation Q.1218 [12] INAP.

4.6.2.2 Procedures for major additions to ETSI INAP

In order to support the introduction of major functional changes, the protocol allows a synchronization between the two applications with regard to which functionality is to be performed. This synchronization takes place before the new function is invoked in either application entity, in order to avoid complicated

fall-back procedures. The solution chosen to achieve such a synchronization is use of the AC negotiation provided in ETS 300 287 [5].

4.6.2.3 Procedures for minor additions to ETSI INAP

The extension mechanism marker shall be used for future standardized minor additions to INAP. This mechanism implements extensions differently by including an "extensions marker" in the type definition. The extensions are expressed by optional fields that are placed after the marker. When an entity receives unrecognized parameters that occur after the marker, they are ignored.

NOTE: Because this version of ASN.1 has not yet been ratified by ITU-T, the marker is placed within a comment for now so that it can easily be uncommented later. As in the ISO case, the extensions are expressed by optional fields that are placed after the marker (which is commented out), in which case they can be ignored.

4.6.2.4 Procedures for inclusion of network specific additions to ETSI INAP

This mechanism is based on the ability to explicitly declare fields of any type via the Macro facility in ASN.1 at the outermost level of a type definition. It works by defining an "ExtensionField" that is placed at the end of the type definition. This extension field is defined as a set of extensions, where an extension can contain any type. Each extension is associated with a value that defines whether the terminating node should ignore the field if unrecognized, or reject the message, similar to the comprehension required mechanism described in the previous subclause. Refer to ITU-T Recommendation Q.1400 [13] for a definition of this mechanism.

5 Single/Multiple Association Control Function (SACF/MACF) rules

5.1 Reflection of TCAP Application Context (AC)

TCAP AC negotiation rules require that the proposed AC, if acceptable, is reflected in the first backwards message.

If the AC is not acceptable, and the TC-User does not wish to continue the dialogue, it may provide an alternate AC to the initiator which can be used to start a new dialogue.

TCAP AC negotiation applies only to the SCF interfaces. Refer to ETS 300 287 [5] for a more detailed description of the TCAP AC negotiation mechanism.

5.2 Sequential/parallel execution of operations

In some cases, it may be necessary to distinguish whether operations should be performed sequentially or in parallel (synchronized). Operations which may be synchronized are:

- charging operations may be synchronized with any other operation.

The method of indicating that operations are to be synchronized is to include them in the same message. Where it is impossible to execute one of the operations identified above until some other operation has progressed to some extent or finished, the sending PE (usually SCP) can control this by sending the operations in two separate messages.

This method does not imply that all operations sent in the same message should be executed simultaneously, but simply that where it could make sense to do so (in the situations identified above) the operations should be synchronized.

In case of inconsistency between the above mentioned generic rules and the FE-specific rules as specified in Clause 7, the FE-specific rules take precedence over the generic rules.

6 Abstract syntax of the CS1 INAP

This Clause specifies the abstract syntax for the CS1 INAP, using ASN.1 as defined in CCITT Recommendation X.208 [14].

The encoding rules which are applicable to the defined abstract syntax are the Basic Encoding Rules for ASN.1, defined in CCITT Recommendation X.209 [15] with the restrictions as described in ITU-T Recommendation Q.773 [10], § 4.1.1, modified by ETS 300 287 [5]. Additional encodings are cited for parameters used in existing ISUP (ETS 300 356-1 [7]) and DSS1 (ETS 300 403-1 [8]) standards.

Any tags and values used in the ITU-T Recommendation Q.1218 [12] and not used in this ETS shall be regarded as reserved.

For the ISUP and DSS1 parameters used in the INAP, only the coding of the parameter value is coded as defined in ISUP or DSS1. The DSS1/ISUP defined parameter identifiers are removed and replaced by the INAP defined parameter identifiers.

The mapping of OPERATION and ERROR to TCAP components is defined in ITU-T Recommendation Q.773 [10] modified by ETS 300 287 [5]. The class of an operation is not stated explicitly but is specified in the ASN.1 OPERATION MACRO, as follows:

class 1: both RESULT and ERRORS appear in the ASN.1 OPERATION MACRO definition;

class 2: only ERRORS appears in the ASN.1 OPERATION MACRO definition;

class 3: only RESULT appears in the ASN.1 OPERATION MACRO definition;

class 4: neither RESULT nor ERRORS appears in the ASN.1 OPERATION MACRO definition.

These map to the classes 2 through 5, respectively, specified in CCITT Recommendation X.219 [16] and ETS 300 196-1 [4] (ITU-T Recommendation Q.932).

The abstract syntax for INAP is composed of several ASN.1 modules describing operations, errors, and associated data types. The values (operation codes and error codes) are defined in a separate module.

The module containing all the type definitions for INAP operations is **IN-CS1-Operations** and is described in subclause 6.1.

The module containing all the type definitions for INAP errors is **IN-CS1-Errors** and is described in subclause 6.2.

The module containing all the type definitions for INAP data types is **IN-CS1-DataTypes** and is described in subclause 6.3.

The module containing the operation codes and error codes for INAP is **IN-CS1-Codes** and is described in subclause 6.4.

All the AC definitions for core INAP are described in subclause 6.5.

6.1 IN CS1 operation types

```
Core-INAP-CS1-Operations {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) modules(0)
cs1-operations(0) version1(0)}

--  This module contains the type definitions for the IN CS1 operations.

DEFINITIONS ::=

BEGIN

IMPORTS

        OPERATION

FROM TCAPMessages {ccitt recommendation q 773 modules(2) messages(1) version2(2)}

--  error types
        Cancelled,
        CancelFailed,
        ETCFailed,
        ImproperCallerResponse,
        MissingCustomerRecord,
        MissingParameter,
        ParameterOutOfRange,
        RequestedInfoError,
        SystemFailure,
        TaskRefused,
        UnavailableResource,
        UnexpectedComponentSequence,
        UnexpectedDataValue,
        UnexpectedParameter,
        UnknownLegID

FROM Core-INAP-CS1-Errors {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) modules(0)
cs1-errors(1) version1(0)}

--  argument types
        ActivateServiceFilteringArg,
        ApplyChargingArg,
        ApplyChargingReportArg,
        AssistRequestInstructionsArg,
        CallGapArg,
        CallInformationReportArg,
        CallInformationRequestArg,
        CancelArg,
        CollectInformationArg,
        ConnectArg,
        ConnectToResourceArg,
        EstablishTemporaryConnectionArg,
        EventNotificationChargingArg,
        EventReportBCSMArg,
        FurnishChargingInformationArg,
        InitialDPArg,
        InitiateCallAttemptArg,
        PlayAnnouncementArg,
        PromptAndCollectUserInformationArg,
        ReceivedInformationArg,
        ReleaseCallArg,
        RequestNotificationChargingEventArg,
        RequestReportBCSMEventArg,
        ResetTimerArg,
        SendChargingInformationArg,
        ServiceFilteringResponseArg,
        SpecializedResourceReportArg

FROM Core-INAP-CS1-DataTypes {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1)
modules(0) cs1-datatypes(2) version1(0)};
```

```
--  TYPE DEFINITIONS FOR IN CS1 OPERATIONS FOLLOW

--  SCF-SSF operations

ActivateServiceFiltering                ::= OPERATION
        ARGUMENT
                ActivateServiceFilteringArg
        RESULT
        ERRORS {
                MissingParameter,
                ParameterOutOfRange,
                SystemFailure,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedParameter
                }

--  Direction: SCF -> SSF, Timer: Tasf
--  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.

ActivityTest                            ::= OPERATION
        RESULT

--  Direction: SCF -> SSF, Timer: Tat
--  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.

ApplyCharging                           ::= OPERATION
        ARGUMENT
                ApplyChargingArg
        ERRORS {
                MissingParameter,
                UnexpectedComponentSequence,
                UnexpectedParameter,
                UnexpectedDataValue,
                ParameterOutOfRange,
                SystemFailure,
                TaskRefused
                }

--  Direction: SCF -> SSF, Timer: Tac
--  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.

ApplyChargingReport                     ::= OPERATION
        ARGUMENT
                ApplyChargingReportArg
        ERRORS {
                MissingParameter,
                UnexpectedComponentSequence,
                UnexpectedParameter,
                UnexpectedDataValue,
                ParameterOutOfRange,
                SystemFailure,
                TaskRefused
                }

--  Direction: SSF -> SCF, Timer: Tacr
--  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.

AssistRequestInstructions               ::= OPERATION
        ARGUMENT
                AssistRequestInstructionsArg
        ERRORS {
                MissingCustomerRecord,
                MissingParameter,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }
```

```
--  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 SSF or SRF to the 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 (in case of hand-off) operation from the SCF.

CallGap                                 ::= OPERATION
        ARGUMENT
                CallGapArg

--  Direction: SCF -> SSF, Timer: Tcg
--  This operation is used to request the SSF to reduce the rate at which specific service requests
--  are sent to the SCF.

CallInformationReport                   ::= OPERATION
        ARGUMENT
                CallInformationReportArg

--  Direction: SSF -> SCF, Timer: Tcirp
--  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.

CallInformationRequest                  ::= OPERATION
        ARGUMENT
                CallInformationRequestArg
        ERRORS {
                MissingParameter,
                ParameterOutOfRange,
                RequestedInfoError,
                SystemFailure,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedParameter
                }

--  Direction: SCF -> SSF, Timer: Tcirg
--  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).

Cancel                                  ::= OPERATION
        ARGUMENT
                CancelArg
        ERRORS {
                CancelFailed
                }

--  Direction:  SCF -> SRF or SCF -> SSF, Timer: Tcan
--  This generic operation cancels the correlated previous operation or all previous requests. The
--  following operations can be cancelled:
--  PlayAnnouncement and PromptAndCollectUserInformation.

CollectInformation                      ::= OPERATION
        ARGUMENT
                CollectInformationArg
        ERRORS {
                MissingParameter,
                SystemFailure,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }

--  Direction: SCF -> SSF, Timer: Tci
--  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).

Connect                                 ::= OPERATION
        ARGUMENT
                ConnectArg
        ERRORS {
                MissingParameter,
                SystemFailure,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }
```

```
--  Direction: SCF -> SSF, Timer: Tcon
--  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., dialled digits) and existing call setup information (e.g.,
--  route index to a list of trunk groups), depending on the information provided by the SCF.

ConnectToResource                       ::= OPERATION
        ARGUMENT
                ConnectToResourceArg
        ERRORS {
                MissingParameter,
                SystemFailure,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }

--  Direction: SCF -> SSF, Timer: Tctr
--  This operation is used to connect a call from the SSP to the PE containing the SRF.

Continue                                ::= OPERATION

--  Direction: SCF -> SSF, Timer: Tcue
--  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.

DisconnectForwardConnection             ::= OPERATION
        ERRORS {
                SystemFailure,
                TaskRefused,
                UnexpectedComponentSequence
                }

--  Direction: SCF -> SSF, Timer: Tdfc
--  This operation is used to disconnect a forward temporary connection  or a connection to a resource.

EstablishTemporaryConnection            ::= OPERATION
        ARGUMENT
                EstablishTemporaryConnectionArg
        ERRORS {
                ETCFailed,
                MissingParameter,
                SystemFailure,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }

--  Direction: SCF -> SSF, Timer: Tetc
--  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.

EventNotificationCharging               ::= OPERATION
        ARGUMENT
                EventNotificationChargingArg

--  Direction: SSF -> SCF, Timer: Tenc
--  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.
--  The operation supports the capabilities to cope with the interactions concerning charging (refer
--  to Annex B, Clause B.5).

EventReportBCSM                         ::= OPERATION
        ARGUMENT
                EventReportBCSMArg

--  Direction: SSF -> SCF, Timer: Terb
--  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.
```

```
FurnishChargingInformation              ::= OPERATION
        ARGUMENT
                FurnishChargingInformationArg
        ERRORS {
                MissingParameter,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }

--  Direction: SCF -> SSF, Timer: Tfci
--  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. The charging scenarios supported by this operation are: 2.2, 2.3 and 2.4 (refer to Annex B
--  where these are defined).

InitialDP                               ::= OPERATION
        ARGUMENT
                InitialDPArg
        ERRORS {
                MissingCustomerRecord,
                MissingParameter,
                SystemFailure,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }

--  Direction: SSF -> SCF, Timer: Tidp
--  This operation is used after a TDP to indicate request for service.

InitiateCallAttempt                     ::= OPERATION
        ARGUMENT
                InitiateCallAttemptArg
        ERRORS {
                MissingParameter,
                SystemFailure,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }

--  Direction: SCF -> SSF, Timer: Tica
--  This operation is used to request the SSF to create a new call to one call party using
--  address information provided by the SCF.

ReleaseCall                             ::= OPERATION
        ARGUMENT
                ReleaseCallArg

--  Direction: SCF -> SSF, Timer: Trc
--  This operation is used to tear down an existing call at any phase of the call for all
--  parties involved in the call.

RequestNotificationChargingEvent        ::= OPERATION
        ARGUMENT
                RequestNotificationChargingEventArg
        ERRORS {
                MissingParameter,
                SystemFailure,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }

--  Direction: SCF -> SSF, Timer: Trnc
--  This operation is used by the SCF to instruct the SSF on how to manage the charging events
--  which are received from other FEs and not under control of the service logic instance. The
--  operation supports the capabilities to cope with the interactions concerning charging (refer to
--  Annex B, Clause B.5).
```

```
RequestReportBCSMEvent                  ::= OPERATION
        ARGUMENT
                RequestReportBCSMEventArg
        ERRORS {
                MissingParameter,
                SystemFailure,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }

--  Direction: SCF -> SSF, Timer: Trrb
--  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.

ResetTimer                              ::= OPERATION
        ARGUMENT
                ResetTimerArg
        ERRORS {
                MissingParameter,
                TaskRefused,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }

--  Direction: SCF -> SSF, Timer: Trt
--  This operation is used to request the SSF to refresh an application timer in the SSF.

SendChargingInformation                 ::= OPERATION
        ARGUMENT
                SendChargingInformationArg
        ERRORS {
                MissingParameter,
                UnexpectedComponentSequence,
                UnexpectedParameter,
                ParameterOutOfRange,
                SystemFailure,
                TaskRefused,
                UnknownLegID
                }

--  Direction: SCF -> SSF, Timer: Tsci
--  This operation is used to  instruct the SSF on the charging information to be sent 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. The charging scenario supported by this operation is
--  scenario 3.2 (refer to Annex B where these are defined).

ServiceFilteringResponse                ::= OPERATION
        ARGUMENT
                ServiceFilteringResponseArg

--  Direction: SSF -> SCF, Timer: Tsfr
--  This operation is used to send back to the SCF the values of counters specified in a previous
--  ActivateServiceFiltering operation.


--  SCF-SRF operations

--  AssistRequestInstructions
--  SRF -> SCF
--  Refer to previous description of this operation in the SCF-SSF operations subclause.

--  Cancel
--  SCF -> SRF
--  Refer to previous description of this operation in the SCF-SSF operations subclause.
```

```
PlayAnnouncement                        ::= OPERATION
        ARGUMENT
                PlayAnnouncementArg
        ERRORS {
                Cancelled,
                MissingParameter,
                SystemFailure,
                UnavailableResource,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }
        LINKED {
                SpecializedResourceReport
                }

--  Direction: SCF -> SRF, Timer: Tpa
--  This operation is to be used after Establish Temporary Connection (assist procedure with a
--  second SSP) or a Connect to Resource (no assist) operation. It may be used for inband
--  interaction with an analogue user, or for interaction with an ISDN user. In the former case, the SRF
--  is usually collocated with the SSF for standard tones (congestion tone etc.) or standard
--  announcements. In the latter case, the SRF is always collocated with the SSF in the switch. Any
--  error is returned to the SCF. The timer associated with this operation must be of a sufficient
--  duration to allow its linked operation to be correctly correlated.

PromptAndCollectUserInformation         ::= OPERATION
        ARGUMENT
                PromptAndCollectUserInformationArg
        RESULT
                ReceivedInformationArg
        ERRORS {
                Cancelled,
                ImproperCallerResponse,
                MissingParameter,
                SystemFailure,
                TaskRefused,
                UnavailableResource,
                UnexpectedComponentSequence,
                UnexpectedDataValue,
                UnexpectedParameter
                }

--  Direction: SCF -> SRF, Timer: Tpc
--  This operation is used to interact with a user to collect information.

SpecializedResourceReport               ::= OPERATION
        ARGUMENT
                SpecializedResourceReportArg

--  Direction: SRF -> SCF, Timer: Tsrr
--  This operation is used as the response to a PlayAnnouncement operation when the announcement completed
--  report indication is set.

END
```

Operation timers

The following value ranges apply for operation specific timers in INAP:

short: 1 to 10 seconds;
medium: 1 to 60 seconds;
long: 1 second to 30 minutes.

Table 3 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 3

Operation Name	Timer	value range
ActivateServiceFiltering	T_{asf}	medium
ActivityTest	T_{at}	short
ApplyCharging	T_{ac}	short
ApplyChargingReport	T_{acr}	short
AssistRequestInstructions	T_{ari}	short
CallGap	T_{cg}	short
CallInformationReport	T_{cirp}	short
CallInformationRequest	T_{cirq}	short
Cancel	T_{can}	short
CollectInformation	T_{ci}	medium
Connect	T_{con}	short
ConnectToResource	T_{ctr}	short
Continue	T_{cue}	short
DisconnectForwardConnection	T_{dfc}	short
EstablishTemporaryConnection	T_{etc}	medium
EventNotificationCharging	T_{enc}	short
EventReportBCSM	T_{erb}	short
FurnishChargingInformation	T_{fci}	short
InitialDP	T_{idp}	short
InitiateCallAttempt	T_{ica}	short
ReleaseCall	T_{rc}	short
RequestNotificationChargingEvent	T_{rnc}	short
RequestReportBCSMEvent	T_{rrb}	short
ResetTimer	T_{rt}	short
SendChargingInformation	T_{sci}	short
ServiceFilteringResponse	T_{sfr}	short
PlayAnnouncement	T_{pa}	long
PromptAndCollectUserInformation	T_{pc}	long
SpecializedResourceReport	T_{srr}	short

6.2 IN CS1 error types

```
Core-INAP-CS1-Errors { ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) modules(0)
cs1-errors(1) version1(0)}

--  This module contains the type definitions for the IN CS1 errors.

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

IMPORTS

        ERROR

FROM TCAPMessages {ccitt recommendation q 773 modules(2) messages(1) version2(2)}

        InvokeID,
        UnavailableNetworkResource

FROM Core-INAP-CS1-DataTypes {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1)
modules(0) cs1-datatypes(2) version1(0)};


--  TYPE DEFINITIONS FOR IN CS1 ERRORS FOLLOW

Cancelled                               ::= ERROR

--  The operation has been cancelled.

CancelFailed                            ::= ERROR
        PARAMETER       SEQUENCE {
                problem         [0] ENUMERATED {
                                        unknownOperation(0),
                                        tooLate(1),
                                        operationNotCancellable(2)
                },
                operation       [1] InvokeID
        }
--  The operation failed to be cancelled.

ETCFailed                               ::= ERROR

--  The establish temporary connection failed.

ImproperCallerResponse                  ::= ERROR

--  The caller response was not as expected.

MissingCustomerRecord                   ::= ERROR

--  The Service Logic Program (SLP) could not be found in the SCF.

MissingParameter                        ::= ERROR

--  An expected optional parameter was not received.

ParameterOutOfRange                     ::= ERROR

--  The parameter was not as expected (e.g., missing or out of range).

RequestedInfoError                      ::= ERROR
        PARAMETER       ENUMERATED {
                unknownRequestedInfo(1),
                requestedInfoNotAvailable(2)
                }

--  The requested information cannot be found.

SystemFailure                           ::= ERROR
        PARAMETER
                UnavailableNetworkResource

--  The operation could not be completed due to a system failure at the serving PE.
```

```
TaskRefused                             ::= ERROR
        PARAMETER       ENUMERATED {
                generic(0),
                unobtainable(1),
                congestion(2)
                }

--  An entity normally capable of the task requested cannot or chooses not to perform the task at
--  this time (this includes error situations like congestion and unobtainable address as used in
--  e.g., the connect operation).

UnavailableResource                     ::= ERROR

--  A requested resource is not available at the serving entity.

UnexpectedComponentSequence             ::= ERROR

--  An incorrect sequence of Components was received (e.g., "DisconnectForwardConnection"
--  followed by "PlayAnnouncement").

UnexpectedDataValue                     ::= ERROR

--  The data value was not as expected (e.g., routing number expected but billing number
--  received)

UnexpectedParameter                     ::= ERROR

--  A parameter received was not expected.

UnknownLegID                            ::= ERROR

--  Leg not known to the SSF.

END
```

6.3 IN CS1 data types

```
EXTENSION MACRO ::=

BEGIN

TYPE NOTATION           ::= ExtensionType Criticality
VALUE NOTATION          ::= value(INTEGER)
ExtensionType           ::= "EXTENSION-SYNTAX" type
Criticality             ::= "CRITICALITY" value(CriticalityType)
CriticalityType         ::= ENUMERATED {
                                ignore(0),
                                abort(1)
                                }
END

--  Example of addition of an extension named "SomeNetworkSpecificIndicator" of type
--  BOOLEAN, with criticality "abort" and to be identified as extension number 1
--  Example of definition using the above macro:
--
--  SomeNetworkSpecificIndicator ::= EXTENSION
--  EXTENSION SYNTAX BOOLEAN
--  CRITICALITY abort
--
--  someNetworkSpecificIndicator SomeNetworkSpecificIndicator ::= 1
--
--  Example of transfer syntax, using the ExtensionField datatype as specified in the module
--  below. Assuming the value of the extension is set to TRUE, the extensions parameter
--  becomes a SEQUENCE of type INTEGER ::= 1, criticality ENUMERATED ::= 1 and
--  value [1] IMPLICIT BOOLEAN ::= TRUE.
```

```
Core-INAP-CS1-DataTypes {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) modules(0)
cs1-datatypes(2) version1(0)}

--  This module contains the type definitions for the IN CS1 data types.

--  The following parameters map onto bearer protocol (i.e., ETS 300 403-1 [8] (DSS1) and
--  ETS 300 356-1 [7] (ISUP)) parameters:
--  CalledPartyNumber, BearerCapability, CallingPartyNumber, HighLayerCompatibility,
--  DestinationRoutingAddress, OriginalCalledPartyID, RedirectingPartyID, RedirectionInformation,
--  AccessTransport, CallingPartyCategory, ForwardCallIndicators, LocationNumber,
--  AssistingSSPIPRoutingAddress, AlertingPattern (Q.931 only), ReleaseCause
--  (and other Cause parameters), AdditionalCallingPartyNumber.

--  The following SSF parameters do not map onto bearer protocol (i.e., ETS 300 403-1 [8] (DSS1)
--  and ETS 300 356-1 [7] (ISUP)) parameters and therefore are assumed to be local to the
--  switching system:
--  RouteList, LegID, IPSSPCapabilities, IPAvailable, CGEncountered,
--  CorrelationID, Timers, MiscCallInfo, and ServiceKey.

--  Where possible, administrations should specify within their network the maximum size of
--  parameters specified in this ETS that are of an indeterminate length.

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

IMPORTS

        InvokeIdType

FROM TCAPMessages {ccitt recommendation q 773 modules(2) messages(1) version2(2)};


--  TYPE DEFINITIONS FOR IN CS1 DATA TYPES FOLLOW

--  Argument Data Types

--  The ordering of parameters in the argument sequences has been arbitrary. Further study may
--  be required to order arguments in a manner which will facilitate efficient encoding and decoding.

ActivateServiceFilteringArg             ::= SEQUENCE {
        filteredCallTreatment    [0] FilteredCallTreatment,
        filteringCharacteristics [1] FilteringCharacteristics,
        filteringTimeOut         [2] FilteringTimeOut,
        filteringCriteria        [3] FilteringCriteria,
        startTime                [4] DateAndTime                                        OPTIONAL,
        extensions               [5] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField   OPTIONAL
--  ...
        }

ApplyChargingArg                        ::= SEQUENCE {
        aChBillingChargingCharacteristics[0] AChBillingChargingCharacteristics,
        sendCalculationToSCPIndication   [1] BOOLEAN      DEFAULT FALSE,
        partyToCharge                    [2] LegID        OPTIONAL,
        extensions                       [3] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField
                                             OPTIONAL
--  ...
        }

--  The sendCalculationToSCPIndication parameter indicates that ApplyChargingReport operations
--  are expected from the SSF. This parameter shall always be set to TRUE.
--  The PartyToCharge parameter indicates the party in the call to which the ApplyCharging
--  operation should be applied. If it is not present, then it is applied to the A-party.

ApplyChargingReportArg                  ::= CallResult

AssistRequestInstructionsArg            ::= SEQUENCE {
        correlationID            [0] CorrelationID,
        iPAvailable              [1] IPAvailable                                        OPTIONAL,
        iPSSPCapabilities        [2] IPSSPCapabilities                                  OPTIONAL,
        extensions               [3] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField   OPTIONAL
--  ...
        }

--  OPTIONAL denotes network operator specific use. The value of the correlationID may be the
--  Called Party Number supplied by the initiating SSF.
```

```
CallGapArg                              ::= SEQUENCE {
        gapCriteria     [0] GapCriteria,
        gapIndicators   [1] GapIndicators,
        controlType     [2] ControlType                                         OPTIONAL,
        gapTreatment    [3] GapTreatment                                        OPTIONAL,
        extensions      [4] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField   OPTIONAL
--  ...
        }

--  OPTIONAL denotes network operator optional. If gapTreatment is not present, the SSF will use
--  a default treatment depending on network operator implementation.

CallInformationReportArg                ::= SEQUENCE {
        requestedInformationList [0] RequestedInformationList,
        extensions              [2] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField   OPTIONAL
--  ...
        }

CallInformationRequestArg               ::= SEQUENCE {
        requestedInformationTypeList    [0] RequestedInformationTypeList,
        extensions                      [2] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField   OPTIONAL
--  ...
        }

CancelArg                               ::= CHOICE {
        invokeID        [0] InvokeID,
        allRequests     [1] NULL
        }

--  The InvokeID has the same value as that which was used for the operation to be cancelled.

CollectInformationArg                   ::= SEQUENCE {
        extensions      [4] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField   OPTIONAL
--  ...
        }

ConnectArg                              ::= SEQUENCE {
        destinationRoutingAddress       [0] DestinationRoutingAddress,
        alertingPattern                 [1] AlertingPattern                     OPTIONAL,
        correlationID                   [2] CorrelationID                       OPTIONAL,
        cutAndPaste                     [3] CutAndPaste                         OPTIONAL,
        originalCalledPartyID           [6] OriginalCalledPartyID               OPTIONAL,
        routeList                       [7] RouteList                           OPTIONAL,
        scfID                           [8] ScfID                               OPTIONAL,
        extensions                      [10] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField
                                                                                OPTIONAL,
        serviceInteractionIndicators    [26] ServiceInteractionIndicators       OPTIONAL,
        callingPartyNumber              [27] CallingPartyNumber                 OPTIONAL,
        callingPartysCategory           [28] CallingPartysCategory              OPTIONAL,
        redirectingPartyID              [29] RedirectingPartyID                 OPTIONAL,
        redirectionInformation          [30] RedirectionInformation             OPTIONAL
--  ...
        }

--  For alerting pattern, OPTIONAL denotes that this parameter only applies if SSF is the
--  terminating local exchange for the subscriber.

ConnectToResourceArg                    ::= SEQUENCE {
        resourceAddress                 CHOICE {
                ipRoutingAddress                [0] IPRoutingAddress,
                none                            [3] NULL
                },
        extensions                      [4] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField
                                                                                OPTIONAL,
        serviceInteractionIndicators    [30] ServiceInteractionIndicators       OPTIONAL
--  ...
        }

EstablishTemporaryConnectionArg         ::= SEQUENCE {
        assistingSSPIPRoutingAddress    [0] AssistingSSPIPRoutingAddress,
        correlationID                   [1] CorrelationID                       OPTIONAL,
        scfID                           [3] ScfID                               OPTIONAL,
        extensions                      [4] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField
                                                                                OPTIONAL,
        serviceInteractionIndicators    [30] ServiceInteractionIndicators       OPTIONAL
--  ...
        }
```

```
EventNotificationChargingArg            ::= SEQUENCE {
        eventTypeCharging               [0] EventTypeCharging,
        eventSpecificInformationCharging [1] EventSpecificInformationCharging          OPTIONAL,
        legID                           [2] LegID                                      OPTIONAL,
        extensions                      [3] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField
                                                                                       OPTIONAL,
        monitorMode                     [30] MonitorMode        DEFAULT notifyAndContinue
--  ...
        }

--  OPTIONAL denotes network operator specific use.

EventReportBCSMArg                      ::= SEQUENCE {
        eventTypeBCSM                   [0] EventTypeBCSM,
        eventSpecificInformationBCSM    [2] EventSpecificInformationBCSM  OPTIONAL,
        legID                           [3] LegID                         OPTIONAL,
        miscCallInfo                    [4] MiscCallInfo        DEFAULT {messageType request},
        extensions                      [5] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField
                                                                          OPTIONAL
--  ...
        }

FurnishChargingInformationArg           ::= FCIBillingChargingCharacteristics

InitialDPArg                            ::= SEQUENCE {
        serviceKey                      [0] ServiceKey,
        calledPartyNumber               [2] CalledPartyNumber           OPTIONAL,
        callingPartyNumber              [3] CallingPartyNumber          OPTIONAL,
        callingPartysCategory           [5] CallingPartysCategory       OPTIONAL,
        cGEncountered                   [7] CGEncountered               OPTIONAL,
        iPSSPCapabilities               [8] IPSSPCapabilities           OPTIONAL,
        iPAvailable                     [9] IPAvailable                 OPTIONAL,
        locationNumber                  [10] LocationNumber             OPTIONAL,
        originalCalledPartyID           [12] OriginalCalledPartyID      OPTIONAL,
        extensions                      [15] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField
                                                                        OPTIONAL,
        highLayerCompatibility          [23] HighLayerCompatibility     OPTIONAL,
        serviceInteractionIndicators    [24] ServiceInteractionIndicators OPTIONAL,
        additionalCallingPartyNumber    [25] AdditionalCallingPartyNumber OPTIONAL,
        forwardCallIndicators           [26] ForwardCallIndicators      OPTIONAL,
        bearerCapability                [27] BearerCapability           OPTIONAL,
        eventTypeBCSM                   [28] EventTypeBCSM              OPTIONAL,
        redirectingPartyID              [29] RedirectingPartyID         OPTIONAL,
        redirectionInformation          [30] RedirectionInformation     OPTIONAL
--  ...
        }

--  OPTIONAL for iPSSPCapabilities, iPAvailable, cGEncountered denotes network operator specific use.
--  OPTIONAL for callingPartyNumber, and callingPartysCategory refer to Clause 7 for the trigger detection
--  point processing rules to specify when these parameters are included in the message.
--  The following parameters shall be recognized by the SCF upon reception of InitialDP:
--      dialledDigits                   [1] CalledPartyNumber           OPTIONAL,
--      callingPartyBusinessGroupID     [4] CallingPartyBusinessGroupID OPTIONAL,
--      callingPartySubaddress          [6] CallingPartySubaddress      OPTIONAL,
--      miscCallInfo                    [11] MiscCallInfo               OPTIONAL,
--      serviceProfileIdentifier        [13] ServiceProfileIdentifier   OPTIONAL,
--      terminalType                    [14] TerminalType               OPTIONAL
--  These parameters shall be ignored by the SCF and not lead to any error procedures.
--  These parameters shall not be sent by a SSF following this ETS.
--  For details on the coding of these parameters refer to ITU-T Recommendation Q.1218 [12].

InitiateCallAttemptArg                  ::= SEQUENCE {
        destinationRoutingAddress       [0] DestinationRoutingAddress,
        alertingPattern                 [1] AlertingPattern             OPTIONAL,
        extensions                      [4] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField
                                                                        OPTIONAL,
        serviceInteractionIndicators    [29] ServiceInteractionIndicators OPTIONAL,
        callingPartyNumber              [30] CallingPartyNumber         OPTIONAL
--  ...
        }

PlayAnnouncementArg                     ::= SEQUENCE {
        informationToSend               [0] InformationToSend,
        disconnectFromIPForbidden       [1] BOOLEAN     DEFAULT TRUE,
        requestAnnouncementComplete     [2] BOOLEAN     DEFAULT TRUE,
        extensions                      [3] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField
                                                        OPTIONAL
--  ...
        }
```

```
PromptAndCollectUserInformationArg        ::= SEQUENCE {
        collectedInfo                     [0] CollectedInfo,
        disconnectFromIPForbidden         [1] BOOLEAN     DEFAULT TRUE,
        informationToSend                 [2] InformationToSend    OPTIONAL,
        extensions                        [3] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField
                                                                   OPTIONAL
--  ...
        }

ReceivedInformationArg                    ::= CHOICE {
        digitsResponse   [0] Digits
        }

ReleaseCallArg                            ::= Cause

--  A default value of decimal 31 (normal unspecified) should be coded appropriately.

RequestNotificationChargingEventArg       ::= SEQUENCE SIZE (1..numOfChargingEvents) OF ChargingEvent

RequestReportBCSMEventArg                 ::= SEQUENCE {
        bcsmEvents       [0] SEQUENCE SIZE (1..numOfBCSMEvents) OF BCSMEvent,
        extensions       [2] SEQUENCE SIZE (1..numOfExtensions) OF ExtensionField  OPTIONAL
--  ...
        }

--  Indicates the BCSM related events for notification.

ResetTimerArg    ::= SEQUENCE {
        timerID [0] TimerID      DEFAULT tssf,
        timervalue       [1] TimerValue,
        extensions       [2] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField   OPTIONAL
        -- ...
        }

SendChargingInformationArg                ::= SEQUENCE {
        sCIBillingChargingCharacteristics[0] SCIBillingChargingCharacteristics,
        legID                             [1] LegID,
        extensions                        [2] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField   OPTIONAL
--  ...
        }

ServiceFilteringResponseArg               ::= SEQUENCE {
        countersValue            [0] CountersValue,
        filteringCriteria        [1] FilteringCriteria    ,
        extensions               [2] SEQUENCE SIZE(1..numOfExtensions) OF ExtensionField   OPTIONAL
--  ...
        }

SpecializedResourceReportArg              ::= NULL


--  Common Data Types

AChBillingChargingCharacteristics         ::= OCTET STRING (SIZE(minAChBillingChargingLength ..
                                                             maxAChBillingChargingLength))

--  The AChBillingChargingCharacteristics 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 with the ApplyChargingReport 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.

AdditionalCallingPartyNumber              ::= Digits

--  Indicates the Additional Calling Party Number.

AlertingPattern                           ::= OCTET STRING (SIZE (3))

--  Indicates a specific pattern that is used to alert a subscriber (e.g., distinctive ringing, tones,
--  etc.). Only applies if SSF is the terminating local exchange for the subscriber. Refer to the
--  ETS 300 403-1 [8] Signal parameter for encoding.

ApplicationTimer                          ::= INTEGER (0..2047)

--  Used by the SCF to set a timer in the SSF. The timer is in seconds.
```

```
AssistingSSPIPRoutingAddress ::= Digits

--  Indicates the destination address of the SRF for the assist procedure.

BCSMEvent                               ::= SEQUENCE {
        eventTypeBCSM           [0] EventTypeBCSM,
        monitorMode             [1] MonitorMode,
        legID                   [2] LegID               OPTIONAL,
        dPSpecificCriteria      [30] DPSpecificCriteria  OPTIONAL
        }

--  Indicates the BCSM Event information for monitoring.

BearerCapability                        ::= CHOICE {
        bearerCap       [0] OCTET STRING (SIZE (2..maxBearerCapabilityLength))
        }

--  Indicates the type of bearer capability connection to the user. For bearerCap, the value as described in
--  DSS1 (ETS 300 403-1 [8])/ISUP (ETS 300 356-1 [7], User Service Information) shall be used.

CalledPartyNumber                       ::= OCTET STRING (SIZE (minCalledPartyNumberLength ..
                                                        maxCalledPartyNumberLength))

--  Indicates the Called Party Number. Refer to ETS 300 356-1 [7] for encoding.

CallingPartyNumber                      ::= OCTET STRING (SIZE (minCallingPartyNumberLength ..
                                                        maxCallingPartyNumberLength))

--  Indicates the Calling Party Number. Refer to ETS 300 356-1 [7] for encoding.

CallingPartysCategory                   ::= OCTET STRING (SIZE (1))

--  Indicates the type of calling party (e.g. operator, payphone, ordinary subscriber). Refer to
--  ETS 300 356-1 [7] for encoding.

CallResult                              ::= OCTET STRING (SIZE (minCallResultLength ..
                                                        maxCallResultLength))

--  This parameter provides the SCF 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 remaining content is network
--  operator specific. 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.

Cause                                   ::= OCTET STRING (SIZE (minCauseLength .. maxCauseLength))

--  Indicates the cause for interface related information. Refer to the ETS 300 356-1 [7] Cause parameter
--  for encoding. For the use of Cause and Location values refer to Q.850.

CGEncountered                           ::= ENUMERATED {
        manualCGencountered(1),
        scpOverload(2)
        }

--  Indicates the type of automatic call gapping encountered, if any.

ChargingEvent                           ::= SEQUENCE {
        eventTypeCharging       [0] EventTypeCharging,
        monitorMode             [1] MonitorMode,
        legID                   [2] LegID       OPTIONAL
        }

--  This parameter indicates the charging event type and corresponding monitor mode and LegID.

CollectedDigits                         ::= SEQUENCE {
        minimumNbOfDigits       [0] INTEGER (1..127)            DEFAULT 1,
        maximumNbOfDigits       [1] INTEGER (1..127),
        endOfReplyDigit         [2] OCTET STRING (SIZE (1..2))  OPTIONAL,
        cancelDigit             [3] OCTET STRING (SIZE (1..2))  OPTIONAL,
        startDigit              [4] OCTET STRING (SIZE (1..2))  OPTIONAL,
        firstDigitTimeOut       [5] INTEGER (1..127)            OPTIONAL,
        interDigitTimeOut       [6] INTEGER (1..127)            OPTIONAL,
        errortreatment          [7] ErrorTreatment              DEFAULT stdErrorAndInfo,
        interruptableAnnInd     [8] BOOLEAN                     DEFAULT TRUE,
        voiceInformation        [9] BOOLEAN                     DEFAULT FALSE,
        voiceBack               [10] BOOLEAN                    DEFAULT FALSE
        }
```

```
--  The use of voiceBack is network operator specific. The endOfReplyDigit, cancelDigit, and startDigit
--  parameters have been designated as OCTET STRING, and are to be encoded as BCD, one digit per OCTET only,
--  contained in the four least significant bits of each OCTET. The usage is service dependent.

CollectedInfo                           ::= CHOICE {
        collectedDigits  [0] CollectedDigits
        }

ControlType                             ::= ENUMERATED {
        sCPOverloaded(0),
        manuallyInitiated(1)
        -- other values ffs
        }

CorrelationID                           ::= Digits

--  used by SCF for correlation with a previous operation. Refer to Clause 7 for a
--  description of the procedures associated with this parameter.

CounterAndValue                         ::= SEQUENCE {
        counterID       [0] CounterID,
        counterValue    [1] Integer4
        }

CounterID                               ::= INTEGER (0..99)

--  Indicates the counters to be incremented. The counterIDs can be addressed by using the last digits of
--  the dialed number.

CountersValue                           ::= SEQUENCE SIZE(0..numOfCounters) OF CounterAndValue

CutAndPaste                             ::= INTEGER (0..22)

--  Indicates the number of digits to be deleted. Refer to ITU-T Recommendation Q.1214 [11], § 6.4.2.16, for
--  additional information.

DateAndTime                             ::= OCTET STRING (SIZE(6))

--  Indicates, amongst others, the start time for activate service filtering. Coded as YYMMDDHHMMSS with
--  each digit coded BCD. The first octet contains YY and the remaining items are sequenced following.
--  EXAMPLE:    1993 September 30th, 12:15:01 would be encoded as:
--              Bits            HGFE    DCBA
--              leading octet   3       9
--                              9       0
--                              0       3
--                              2       1
--                              5       1
--                              1       0

DestinationRoutingAddress               ::= SEQUENCE SIZE (1) OF CalledPartyNumber

--  Indicates the Called Party Number.

Digits                                  ::= OCTET STRING (SIZE (minDigitsLength .. maxDigitsLength))

--  Indicates the address signalling digits. Refer to the ETS 300 356-1 [7] Generic Number and Generic
--  Digits parameters for encoding. The coding of the subfields "NumberQualifier" in Generic Number and
--  "Type Of Digits" 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, therefor the value is network
--  operator specific.
--
--  The following parameter should use Generic Number:
--  CorrelationID for AssistRequestInstructions,
--  AdditionalCallingPartyNumber for InitialDP,
--  AssistingSSPIPRoutingAddress for EstablishTemporaryConnection,
--  calledAddressValue for all occurrences,
--  callingAddressValue for all occurrences
--
--  The following parameters should use Generic Digits:
--  all other CorrelationID occurrences,
--  number VariablePart,
--  digitsResponse ReceivedInformationArg

DisplayInformation                      ::= IA5String (SIZE (minDisplayInformationLength ..
                                                             maxDisplayInformationLength))

--  Indicates the display information.
```

```
DPSpecificCriteria                      ::= CHOICE {
        numberOfDigits   [0] NumberOfDigits,
        applicationTimer [1] ApplicationTimer
        }

--  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.

Duration                                ::= INTEGER (-2..86400)

--  Values are seconds. Negative values denote a special value, refer to the procedure description of the
--  relevant operations for further information.

ErrorTreatment                          ::= ENUMERATED {
        stdErrorAndInfo(0),
        help(1),
        repeatPrompt(2)}

--  stdErrorAndInfo means returning the "ImproperCallerResponse" error in the event of an error condition
--  during collection of user info.

EventSpecificInformationBCSM            ::=  CHOICE {
        collectedInfoSpecificInfo       [0] SEQUENCE {
                calledPartyNumber               [0] CalledPartyNumber
                --...--
                },
        analyzedInfoSpecificInfo        [1] SEQUENCE {
                calledPartyNumber               [0] CalledPartyNumber
                --...--
                },
        routeSelectFailureSpecificInfo  [2] SEQUENCE {
                failureCause                    [0] Cause       OPTIONAL
                --...--
                },
        oCalledPartyBusySpecificInfo    [3] SEQUENCE {
                busyCause                       [0] Cause       OPTIONAL
                --...--
                },
        oNoAnswerSpecificInfo           [4] SEQUENCE {
                -- no specific info defined --
                --...--
                },
        oAnswerSpecificInfo             [5] SEQUENCE {
                -- no specific info defined --
                --...--
                },
        oMidCallSpecificInfo            [6] SEQUENCE {
                -- no specific info defined --
                --...--
                },
        oDisconnectSpecificInfo         [7] SEQUENCE {
                releaseCause                    [0] Cause       OPTIONAL
                --...--
                },
        tCalledPartyBusySpecificInfo    [8] SEQUENCE {
                busyCause                       [0] Cause       OPTIONAL
                --...--
                },
        tNoAnswerSpecificInfo           [9] SEQUENCE {
                -- no specific info defined --
                --...--
                },
        tAnswerSpecificInfo             [10] SEQUENCE {
                -- no specific info defined --
                --...--
                },
        tMidCallSpecificInfo            [11] SEQUENCE {
                -- no specific info defined --
                --...--
                },
        tDisconnectSpecificInfo         [12] SEQUENCE {
                releaseCause                    [0] Cause       OPTIONAL
                --...--
                }
        }

--  Indicates the call related information specific to the event.
```

```
EventSpecificInformationCharging          ::= OCTET STRING (SIZE(minEventSpecificInformationChargingLength
                                                          ..maxEventSpecificInformationChargingLength))

--  defined by network operator. Indicates the charging related information specific to the event.
--  An example data type definition for this parameter is given below:
--  EventSpecificInformationCharging      ::= CHOICE {
--       chargePulses     [0] Integer4,
--       chargeMessages   [1] OCTET STRING (SIZE (min..max))
--       }

EventTypeBCSM                             ::= ENUMERATED {
        origAttemptAuthorized(1),
        collectedInfo(2),
        analyzedInformation(3),
        routeSelectFailure(4),
        oCalledPartyBusy(5),
        oNoAnswer(6),
        oAnswer(7),
        oMidCall(8),
        oDisconnect(9),
        oAbandon(10),
        termAttemptAuthorized(12),
        tCalledPartyBusy(13),
        tNoAnswer(14),
        tAnswer(15),
        tMidCall(16),
        tDisconnect(17),
        tAbandon(18)
        }

--  Indicates the BCSM detection point event. Refer to ITU-T Recommendation Q.1214 [11], § 4.2.2.2 for
--  additional information on the events. Values origAttemptAuthorized and termAttemptAuthorized can only be
--  used for TDPs.

EventTypeCharging                         ::= OCTET STRING (SIZE (minEventTypeChargingLength ..
                                                            maxEventTypeChargingLength))

--  This parameter indicates the charging event type. Its contents is network operator specific.
--  An example data type definition for this parameter is given below:
--  EventTypeCharging    ::= ENUMERATED {
--       chargePulses(0),
--       chargeMessages(1)
--       }

ExtensionField                            ::= SEQUENCE {
        type             INTEGER,         --  shall identify the value of an EXTENSION type
        criticality      ENUMERATED {
                ignore(0),
                abort(1)
                } DEFAULT ignore,
        value            [1] ANY DEFINED BY type
        }

--  This parameter indicates an extension of an argument data type. Its contents is network operator
--  specific.

FCIBillingChargingCharacteristics         ::= OCTET STRING (SIZE (minFCIBillingChargingLength ..
                                                            maxFCIBillingChargingLength))

--  This parameter indicates the billing and/or charging characteristics. Its content is network operator
--  specific. An example datatype definition for this parameter is given below:
--  FCIBillingChargingCharacteristics     ::= CHOICE {
--       completeChargingRecord   [0] OCTET STRING (SIZE (min...max)),
--       correlationID            [1] CorrelationID,
--       scenario2Dot3            [2] SEQUENCE {
--               chargeParty              [0] LegID                                  OPTIONAL,
--               chargeLevel              [1] OCTET STRING (SIZE (min..max))         OPTIONAL,
--               chargeItems              [2] SET OF Attribute                       OPTIONAL
--               }
--       }
--  Depending on the applied charging scenario the following information elements can be included (refer to
--  Annex B):
--       complete charging record (scenario 2.2)
--       charge party (scenario 2.3)
--       charge level (scenario 2.3)
--       charge items (scenario 2.3)
--       correlationID (scenario 2.4)
```

```
FilteredCallTreatment                   ::= SEQUENCE {
        sFBillingChargingCharacteristics [0] SFBillingChargingCharacteristics,
        informationToSend                [1] InformationToSend            OPTIONAL,
        maximumNumberOfCounters          [2] MaximumNumberOfCounters      OPTIONAL,
        releaseCause                     [3] Cause                        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 CounterAndValue

FilteringCharacteristics                ::= CHOICE {
        interval        [0] INTEGER (-1..32000),
        numberOfCalls   [1] Integer4
        }

--  Indicates the severity of the filtering and the point in time when the ServiceFilteringResponse is to be
--  sent. If = interval, every interval of time the next call leads to an InitialDP and a
--  ServiceFilteringResponse is sent to the SCF. The interval is specified in seconds.If = NumberOfCalls,
--  every N calls the Nth call leads to an InitialDP and a ServiceFilteringResponse is sent to the SCF. If
--  ActivateServiceFiltering implies several counters (filtering on several dialled numbers), the
--  numberOfCalls would include calls to all the dialled numbers.

FilteringCriteria                       ::= CHOICE {
        serviceKey              [2] ServiceKey,
        addressAndService       [30] SEQUENCE {
                calledAddressValue      [0] Digits,
                serviceKey              [1] ServiceKey,
                callingAddressValue     [2] Digits              OPTIONAL,
                locationNumber          [3] LocationNumber      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.

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 ETS 300 356-1 [7] for encoding.

GapCriteria                             ::= CHOICE {
        calledAddressValue      [0] Digits,
        gapOnService            [2] GapOnService,
        calledAddressAndService  [29] SEQUENCE {
                calledAddressValue      [0] Digits,
                serviceKey              [1] ServiceKey
                },
        callingAddressAndService [30] SEQUENCE {
                callingAddressValue     [0] Digits,
                serviceKey              [1] ServiceKey,
                locationNumber          [2] LocationNumber      OPTIONAL
                }
        }
--  Both calledAddressValue and callingAddressValue can be incomplete numbers, in the sense that a limited
--  amount of digits can be given. For the handling of numbers starting with the same digit string refer to
--  the detailed procedure of the CallGap operation in Clause 9.

GapOnService                            ::= SEQUENCE {
        serviceKey      [0] ServiceKey
        }

GapIndicators                           ::= SEQUENCE {
        duration        [0] Duration,
        gapInterval     [1] Interval
        }

--  Indicates the gapping characteristics. No gapping when gapInterval equals 0, and gap all calls when
--  gapInterval equals -1. For further information regarding the meaning of specific values of duration and
--  gapInterval refer to the detailed procedure of the CallGap operation in Clause 9.
```

```
GapTreatment                            ::= CHOICE {
        informationToSend       [0] InformationToSend,
        releaseCause            [1] Cause,
        both                    [2] SEQUENCE {
                informationToSend       [0] InformationToSend,
                releaseCause            [1] Cause
                }
        }

--  The default value for Cause is the same as in ISUP.

HighLayerCompatibility                  ::= OCTET STRING (SIZE(highLayerCompatibilityLength))

--  Indicates the teleservice. For encoding, DSS1 (ETS 300 403-1 [8]) is used.

InbandInfo                              ::= SEQUENCE {
        messageID               [0] MessageID,
        numberOfRepetitions     [1] INTEGER (1..127)     OPTIONAL,
        duration                [2] INTEGER (0..32767)   OPTIONAL,
        interval                [3] INTEGER (0..32767)   OPTIONAL
        }

--  interval is the time in seconds between each repeated announcement. Duration is the total amount of time
--  in seconds, including repetitions and intervals. The end of announcement is either the end of duration
--  or numberOfRepetitions, whatever comes first. Duration with value 0 indicates infinite duration.

InformationToSend                       ::= CHOICE {
        inbandinfo              [0] InbandInfo,
        tone                    [1] Tone,
        displayInformation      [2] DisplayInformation
        }

Integer4                                ::= INTEGER (0..2147483647)

Interval                                ::= INTEGER (-1..60000)

--  Units are milliseconds. A -1 value denotes infinite.

InvokeID                                ::= InvokeIdType

--  Operation invoke identifier.

IPAvailable                             ::= OCTET STRING (SIZE (minIPAvailableLength ..
                                                        maxIPAvailableLength))

--  defined by network operator. Indicates that the resource is available.

IPRoutingAddress                        ::= CalledPartyNumber

--  Indicates the routing address for the IP.

IPSSPCapabilities                       ::= OCTET STRING (SIZE (minIPSSPCapabilitiesLength ..
                                                        maxIPSSPCapabilitiesLength))

--  defined by network operator. Indicates the SRF resources available at the SSP.

LegID   ::= CHOICE {
        sendingSideID    [0] LegType,  -- used in operations sent from SCF to SSF
        receivingSideID  [1] LegType   -- used in operations sent from SSF to SCF
        }

--  Indicates a reference to a specific party in a call. OPTIONAL denotes network operator specific use with
--  unilateral ID assignment. OPTIONAL for LegID also denotes the following:
--   - when only one party exists in the 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 which party is concerned.
--     2) LegID is not present and a default value is assumed (e.g., calling party in the case of    the
--        ApplyCharging operation).

LegType                                 ::= OCTET STRING (SIZE(1))
        leg1 LegType     ::= '01'H
        leg2 LegType     ::= '02'H

LocationNumber                          ::= OCTET STRING (SIZE (minLocationNumberLength ..
                                                        maxLocationNumberLength))

--  Indicates the Location Number for the calling party. Refer to ETS 300 356-1 [7] for encoding.
```

```
MaximumNumberOfCounters                 ::= INTEGER (1.. numOfCounters)

MessageID                               ::= CHOICE {
        elementaryMessageID      [0] Integer4,
        text                     [1] SEQUENCE {
                messageContent           [0] IA5String (SIZE(minMessageContentLength ..
                                                        maxMessageContentLength)),
                attributes               [1] OCTET STRING (SIZE (minAttributesLength ..
                                                        maxAttributesLength))           OPTIONAL
                },
        elementaryMessageIDs     [29] SEQUENCE SIZE (1..numOfMessageIDs) OF Integer4,
        variableMessage          [30] SEQUENCE {
                elementaryMessageID      [0] Integer4,
                variableParts            [1] SEQUENCE SIZE(1..5) OF VariablePart
                }
        }

--  OPTIONAL denotes network operator specific use.

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. If this parameter is used in the context of
--  charging events, the following definitions apply for the handling of charging events:
--  Interrupted means that the SSF notifies the SCF of the charging event using EventNotificationCharging
--  and does not process the event but discard it.
--  NotifyAndContinue means that SSF notifies the SCF of the charging event using EventNotificationCharging
--  and continues processing the event or signal without waiting for SCF instructions.
--  Transparent means that the SSF does not notify the SCF of the event. This value is used to end the
--  monitoring of a previously requested charging event. Previously requested charging events are monitored
--  until ended by a transparent monitor mode, or until the end of the connection configuration.
--  For the use of this parameter in the context of BCSM events is referred to subclauses 9.17 and 9.25.

NumberOfDigits                          ::= INTEGER(1..255)

--  Indicates the number of digits to be collected.

OriginalCalledPartyID                   ::= OCTET STRING (SIZE (minOriginalCalledPartyIDLength ..
                                                        maxOriginalCalledPartyIDLength))

--  Indicates the original called number. Refer to ETS 300 356-1 [7] Original Called Number for encoding.

RedirectingPartyID                      ::= OCTET STRING (SIZE (minRedirectingPartyIDLength ..
                                                        maxRedirectingPartyIDLength))

--  Indicates redirecting number. Refer to ETS 300 356-1 [7] Redirecting number for encoding.

RedirectionInformation                  ::= OCTET STRING (SIZE (2))

--  Indicates redirection information. Refer to ETS 300 356-1 [7] Redirection Information for encoding.

RequestedInformationList                ::= SEQUENCE SIZE(1..numOfInfoItems) OF RequestedInformation

RequestedInformationTypeList            ::= SEQUENCE SIZE(1..numOfInfoItems) OF RequestedInformationType

RequestedInformation                    ::= SEQUENCE {
        requestedInformationType        [0] RequestedInformationType,
        requestedInformationValue       [1] RequestedInformationValue
        }

RequestedInformationType                ::= ENUMERATED {
        callAttemptElapsedTime(0),
        callStopTime(1),
        callConnectedElapsedTime(2),
        calledAddress(3),
        releaseCause(30)
        }
```

```
RequestedInformationValue               ::= CHOICE {
        callAttemptElapsedTimeValue     [0] INTEGER (0..255),
        callStopTimeValue               [1] DateAndTime,
        callConnectedElapsedTimeValue   [2] Integer4,
        calledAddressValue              [3] Digits,
        releaseCauseValue               [30] Cause
        }

--  The callAttemptElapsedTimeValue is specified in seconds. The unit for the callConnectedElapsedTimeValue
--  is 100 milliseconds.

RouteList                               ::= SEQUENCE SIZE(1..3) OF OCTET STRING (SIZE(minRouteListLength
                                                                                ..maxRouteListLength))

--  Indicates a list of trunk groups or a route index. See ITU-T Recommendation Q.1214 [11] for additional
--  information on this item.

ScfID                                   ::= OCTET STRING (SIZE (minScfIDLength .. maxScfIDLength))

--  defined by network operator. Indicates the SCF identifier.

SCIBillingChargingCharacteristics       ::= OCTET STRING (SIZE (minSCIBillingChargingLength ..
                                                                maxSCIBillingChargingLength))

--  This parameter indicates the billing and/or charging characteristics. Its content is network operator
--  specific. An example datatype definition for this parameter is given below:
--  SCIBillingChargingCharacteristics   ::= CHOICE {
--       chargeLevel      [0] OCTET STRING (SIZE (min..max),
--       chargePulses     [1] Integer4,
--       chargeMessages   [2] OCTET STRING (SIZE(min..max)
--       }
--  Depending on the applied charging scenario, the following information elements can be included (refer to
--  Annex B):
--       chargeLevel (scenario 3.2);
--       chargePulses (scenario 3.2);
--       chargeMessages (scenario 3.2).

ServiceInteractionIndicators            ::= OCTET STRING (SIZE (minServiceInteractionIndicatorsLength ..
                                                                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. The content is network
--  operator specific.

ServiceKey                              ::= Integer4

--  Information that allows the SCF to choose the appropriate service logic.

SFBillingChargingCharacteristics        ::= OCTET STRING (SIZE (minSFBillingChargingLength ..
                                                                maxSFBillingChargingLength))

--  This parameter indicates the billing and/or charging characteristics for filtered calls. Its content is
--  network operator specific.

TimerID                                 ::= ENUMERATED {
        tssf(0)         --  others for further study
        }

--  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.
```

```
UnavailableNetworkResource              ::= ENUMERATED {
        unavailableResources(0),
        componentFailure(1),
        basicCallProcessingException(2),
        resourceStatusFailure(3),
        endUserFailure(4)
        }

--  Indicates the network resource that failed.

VariablePart                            ::= CHOICE {
        integer [0] Integer4,
        number  [1] Digits,                         -- Generic digits
        time    [2] OCTET STRING (SIZE(2)),         -- HH:MM, BCD coded
        date    [3] OCTET STRING (SIZE(3)),         -- YYMMDD, BCD coded
        price   [4] OCTET STRING (SIZE(4))          -- DDDDDD.DD, BCD coded
        }

--  Indicates the variable part of the message. BCD coded variable parts are encoded as described below.
--  EXAMPLE 1:   time = 12:15 would be encoded as:
--               Bits            HGFE    DCBA
--               leading octet   2       1
--                               5       1
--  EXAMPLE 2:   date = 1993 September 30th would be encoded as:
--               Bits            HGFE    DCBA
--               leading octet   3       9
--                               9       0
--                               0       3
--  EXAMPLE 3:   price = ECU 249.50 would be encoded as:
--               Bits            HGFE    DCBA
--               leading octet   0       0
--                               2       0
--                               9       4
--                               0       5

--  Definition of range constants

highLayerCompatibilityLength            INTEGER ::= 2
minAChBillingChargingLength             INTEGER ::= -- network specific
maxAChBillingChargingLength             INTEGER ::= -- network specific
minAttributesLength                     INTEGER ::= -- network specific
maxAttributesLength                     INTEGER ::= -- network specific
maxBearerCapabilityLength               INTEGER ::= -- network specific
minCalledPartyNumberLength              INTEGER ::= -- network specific
maxCalledPartyNumberLength              INTEGER ::= -- network specific
minCallingPartyNumberLength             INTEGER ::= -- network specific
maxCallingPartyNumberLength             INTEGER ::= -- network specific
minCallResultLength                     INTEGER ::= -- network specific
maxCallResultLength                     INTEGER ::= -- network specific
minCauseLength                          INTEGER ::= 2
maxCauseLength                          INTEGER ::= -- network specific
minDigitsLength                         INTEGER ::= -- network specific
maxDigitsLength                         INTEGER ::= -- network specific
minDisplayInformationLength             INTEGER ::= -- network specific
maxDisplayInformationLength             INTEGER ::= -- network specific
minEventSpecificInformationChargingLength INTEGER ::= -- network specific
maxEventSpecificInformationChargingLength INTEGER ::= -- network specific
minEventTypeChargingLength              INTEGER ::= -- network specific
maxEventTypeChargingLength              INTEGER ::= -- network specific
minFCIBillingChargingLength             INTEGER ::= -- network specific
maxFCIBillingChargingLength             INTEGER ::= -- network specific
minIPAvailableLength                    INTEGER ::= -- network specific
maxIPAvailableLength                    INTEGER ::= -- network specific
minIPSSPCapabilitiesLength              INTEGER ::= -- network specific
maxIPSSPCapabilitiesLength              INTEGER ::= -- network specific
minLocationNumberLength                 INTEGER ::= -- network specific
maxLocationNumberLength                 INTEGER ::= -- network specific
minMessageContentLength                 INTEGER ::= -- network specific
maxMessageContentLength                 INTEGER ::= -- network specific
minOriginalCalledPartyIDLength          INTEGER ::= -- network specific
maxOriginalCalledPartyIDLength          INTEGER ::= -- network specific
minRedirectingPartyIDLength             INTEGER ::= -- network specific
maxRedirectingPartyIDLength             INTEGER ::= -- network specific
minRouteListLength                      INTEGER ::= -- network specific
maxRouteListLength                      INTEGER ::= -- network specific
minScfIDLength                          INTEGER ::= -- network specific
maxScfIDLength                          INTEGER ::= -- network specific
minSCIBillingChargingLength             INTEGER ::= -- network specific
maxSCIBillingChargingLength             INTEGER ::= -- network specific
```

```
minServiceInteractionIndicatorsLength     INTEGER ::= -- network specific
maxServiceInteractionIndicatorsLength     INTEGER ::= -- network specific
minSFBillingChargingLength                INTEGER ::= -- network specific
maxSFBillingChargingLength                INTEGER ::= -- network specific
numOfBCSMEvents                           INTEGER ::= -- network specific
numOfChargingEvents                       INTEGER ::= -- network specific
numOfCounters                             INTEGER ::= 100
numOfExtensions                           INTEGER ::= -- network specific
numOfInfoItems                            INTEGER ::= 5
numOfMessageIDs                           INTEGER ::= -- network specific

END
```

6.4 IN CS1 application protocol (operation and error codes)

```
Core-INAP-CS1-Codes {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) modules(0)
cs1-codes(3) version1(0)}

--  This module contains the operation and error code assignments for the IN CS1 application protocol.

DEFINITIONS ::=

BEGIN

--  OPERATION AND ERROR CODE ASSIGNMENTS FOR THE IN CS1 PROTOCOL FOLLOWS

IMPORTS

--  macros
        APPLICATION-SERVICE-ELEMENT

FROM Remote-Operations-Notation-Extension {joint-iso-ccitt remote-operations(4) notation-extension(2)}

--  operation types
        ActivateServiceFiltering,
        ActivityTest,
        ApplyCharging,
        ApplyChargingReport,
        AssistRequestInstructions,
        CallGap,
        CallInformationReport,
        CallInformationRequest,
        Cancel,
        CollectInformation,
        Connect,
        ConnectToResource,
        Continue,
        DisconnectForwardConnection,
        EstablishTemporaryConnection,
        EventNotificationCharging,
        EventReportBCSM,
        FurnishChargingInformation,
        InitialDP,
        InitiateCallAttempt,
        PlayAnnouncement,
        PromptAndCollectUserInformation,
        ReleaseCall,
        RequestNotificationChargingEvent,
        RequestReportBCSMEvent,
        ResetTimer,
        SendChargingInformation,
        ServiceFilteringResponse,
        SpecializedResourceReport

FROM Core-INAP-CS1-Operations { ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1)
modules(0) cs1-operations(0) version1(0)}

--  error types
        Cancelled,
        CancelFailed,
        ETCFailed,
        ImproperCallerResponse,
        MissingCustomerRecord,
        MissingParameter,
        ParameterOutOfRange,
        RequestedInfoError,
        SystemFailure,
```

```
        TaskRefused,
        UnavailableResource,
        UnexpectedComponentSequence,
        UnexpectedDataValue,
        UnexpectedParameter,
        UnknownLegID

FROM Core-INAP-CS1-Errors {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) modules(0)
cs1-errors(1) version1(0)};

--  the operations are grouped by the identified ASEs.

--  SCF activation ASE

initialDP                       InitialDP                       ::= localValue 0

--  SCF/SRF activation of assist ASE

assistRequestInstructions       AssistRequestInstructions       ::= localValue 16

--  Assist connection establishment ASE

establishTemporaryConnection    EstablishTemporaryConnection    ::= localValue 17

--  Generic disconnect resource ASE

disconnectForwardConnection     DisconnectForwardConnection     ::= localValue 18

--  Non-assisted connection establishment ASE

connectToResource               ConnectToResource               ::= localValue 19

--  Connect ASE (elementary SSF function)

connect                         Connect                         ::= localValue 20

--  Call handling ASE (elementary SSF function)

releaseCall                     ReleaseCall                     ::= localValue 22

--  BCSM Event handling ASE

requestReportBCSMEvent          RequestReportBCSMEvent          ::= localValue 23
eventReportBCSM                 EventReportBCSM                 ::= localValue 24

--  Charging Event handling ASE

requestNotificationChargingEvent RequestNotificationChargingEvent ::= localValue 25
eventNotificationCharging       EventNotificationCharging       ::= localValue 26

--  SSF call processing ASE

collectInformation              CollectInformation              ::= localValue 27
continue                        Continue                        ::= localValue 31

--  SCF call initiation ASE

initiateCallAttempt             InitiateCallAttempt             ::= localValue 32

--  Timer ASE

resetTimer                      ResetTimer                      ::= localValue 33

--  Billing ASE

furnishChargingInformation      FurnishChargingInformation      ::= localValue 34

--  Charging ASE

applyCharging                   ApplyCharging                   ::= localValue 35
applyChargingReport             ApplyChargingReport             ::= localValue 36

--  Traffic management ASE

callGap                         CallGap                         ::= localValue 41
```

```
--  Service management ASE

activateServiceFiltering         ActivateServiceFiltering         ::= localValue 42
serviceFilteringResponse         ServiceFilteringResponse         ::= localValue 43

--  Call report ASE

callInformationReport            CallInformationReport            ::= localValue 44
callInformationRequest           CallInformationRequest           ::= localValue 45

--  Signalling control ASE

sendChargingInformation          SendChargingInformation          ::= localValue 46

--  Specialized resource control ASE

playAnnouncement                 PlayAnnouncement                 ::= localValue 47
promptAndCollectUserInformation  PromptAndCollectUserInformation  ::= localValue 48
specializedResourceReport        SpecializedResourceReport        ::= localValue 49

--  Cancel ASE

cancel                           Cancel                           ::= localValue 53

--  Activity Test ASE

activityTest                     ActivityTest                     ::= localValue 55

--  ERROR codes

cancelled                        Cancelled                        ::= localValue 0
cancelFailed                     CancelFailed                     ::= localValue 1
eTCFailed                        ETCFailed                        ::= localValue 3
improperCallerResponse           ImproperCallerResponse           ::= localValue 4
missingCustomerRecord            MissingCustomerRecord            ::= localValue 6
missingParameter                 MissingParameter                 ::= localValue 7
parameterOutOfRange              ParameterOutOfRange              ::= localValue 8
requestedInfoError               RequestedInfoError               ::= localValue 10
systemFailure                    SystemFailure                    ::= localValue 11
taskRefused                      TaskRefused                      ::= localValue 12
unavailableResource              UnavailableResource              ::= localValue 13
unexpectedComponentSequence      UnexpectedComponentSequence      ::= localValue 14
unexpectedDataValue              UnexpectedDataValue              ::= localValue 15
unexpectedParameter              UnexpectedParameter              ::= localValue 16
unknownLegID                     UnknownLegID                     ::= localValue 17

--  APPLICATION SERVICE ELEMENTS

SCF-Activation-ASE                       ::= APPLICATION-SERVICE-ELEMENT
        -- consumer is SSF
        CONSUMER INVOKES {
                initialDP
                }
SCF-SRF-activation-of-assist-ASE         ::= APPLICATION-SERVICE-ELEMENT
        -- consumer is SSF/SRF
        CONSUMER INVOKES {
                assistRequestInstructions
                }
Assist-connection-establishment-ASE      ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                establishTemporaryConnection
                }
Generic-disconnect-resource-ASE          ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                disconnectForwardConnection
                }
Non-assisted-connection-establishment-ASE ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                connectToResource
                }
Connect-ASE                              ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                connect
                }
```

```
Call-handling-ASE                       ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                releaseCall
                }
BCSM-event-handling-ASE                 ::= APPLICATION-SERVICE-ELEMENT
        -- consumer is SSF
        CONSUMER INVOKES {
                eventReportBCSM
                }
        -- supplier is SCF
        SUPPLIER INVOKES {
                requestReportBCSMEvent
                }
Charging-event-handling-ASE             ::= APPLICATION-SERVICE-ELEMENT
        -- consumer is SSF
        CONSUMER INVOKES {
                eventNotificationCharging
                }
        -- supplier is SCF
        SUPPLIER INVOKES {
                requestNotificationChargingEvent
                }
SSF-call-processing-ASE                 ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                collectInformation,
                continue
                }
SCF-call-initiation-ASE                 ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                initiateCallAttempt
                }
Timer-ASE                               ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                resetTimer
                }
Billing-ASE                             ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                furnishChargingInformation
                }
Charging-ASE                            ::= APPLICATION-SERVICE-ELEMENT
        -- consumer is SSF
        CONSUMER INVOKES {
                applyChargingReport
                }
        -- supplier is SCF
        SUPPLIER INVOKES {
                applyCharging
                }
Traffic-management-ASE                  ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                callGap
                }
Service-management-ASE                  ::= APPLICATION-SERVICE-ELEMENT
        -- consumer is SSF
        CONSUMER INVOKES {
                serviceFilteringResponse
                }
        -- supplier is SCF
        SUPPLIER INVOKES {
                activateServiceFiltering
                }
Call-report-ASE                         ::= APPLICATION-SERVICE-ELEMENT
        -- consumer is SSF
        CONSUMER INVOKES {
                callInformationReport
                }
        -- supplier is SCF
        SUPPLIER INVOKES {
                callInformationRequest
                }
```

```
Signalling-control-ASE                  ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                sendChargingInformation
                }
Specialized-resource-control-ASE        ::= APPLICATION-SERVICE-ELEMENT
        -- consumer is SSF/SRF
        CONSUMER INVOKES {
                specializedResourceReport
                }
        -- supplier is SCF
        SUPPLIER INVOKES {
                playAnnouncement,
                promptAndCollectUserInformation
                }
Cancel-ASE                              ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                cancel
                }
Activity-test-ASE                       ::= APPLICATION-SERVICE-ELEMENT
        -- supplier is SCF
        SUPPLIER INVOKES {
                activityTest
                }
END
```

6.5 IN CS1 application contexts

```
APPLICATION-CONTEXT MACRO ::=

BEGIN

TYPE NOTATION           ::= Symmetric | InitiatorConsumerOf ResponderConsumerOf | empty
VALUE NOTATION          ::= value(VALUE OBJECT IDENTIFIER)
Symmetric               ::= "OPERATIONS OF" "{" ASEList "}"
InitiatorConsumerOf     ::= "INITIATOR CONSUMER OF" "{" ASEList "}" | empty
ResponderConsumerOf     ::= "RESPONDER CONSUMER OF" "{" ASEList "}" | empty
ASEList                 ::= ASE | ASEList "," ASE
ASE                     ::= type-- shall reference an APPLICATION-SERVICE-ELEMENT type.

END


Core-INAP-CS1-SSP-to-SCP-AC APPLICATION-CONTEXT
-- dialogue initiated by SSP with InitialDP
INITIATOR CONSUMER OF {
        SCF-activation-ASE,
        Assist-connection-establishment-ASE,
        Generic-disconnect-resource-ASE,
        Non-assisted-connection-establishment-ASE,
        Connect-ASE
        Call-handling-ASE,
        BCSM-event-handling-ASE,
        Charging-event-handling-ASE,
        SSF-call-processing-ASE,
        Timer-ASE,
        Billing-ASE,
        Charging-ASE,
        Traffic-management-ASE,
        Call-report-ASE,
        Signalling-control-ASE,
        Specialized-resource-control-ASE,
        Cancel-ASE,
        Activity-test-ASE
        }
::= {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) ac(1)
cs1-ssp-to-scp(0) version1(0)};
```

```
Core-INAP-CS1-assist-handoff-SSP-to-SCP-AC APPLICATION-CONTEXT
-- dialogue initiated by SSP with AssistRequestInstructions
INITIATOR CONSUMER OF {
        SCF-SRF-activation-of-assist-ASE,
        Generic-disconnect-resource-ASE,
        Non-assisted-connection-establishment-ASE,
        Call-handling-ASE,
        Timer-ASE,
        Billing-ASE,
        Charging-ASE,
        Specialized-resource-control-ASE,
        Cancel-ASE,
        Activity-test-ASE
        }
::= {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) ac(1)
cs1-assist-handoff-ssp-to-scp(1) version1(0)};

Core-INAP-CS1-IP-to-SCP-AC APPLICATION-CONTEXT
-- dialogue initiated by IP with AssistRequestInstructions
INITIATOR CONSUMER OF {
        SCF-SRF-activation-of-assist-ASE,
        Timer-ASE,
        Specialized-resource-control-ASE,
        Cancel-ASE,
        Activity-test-ASE
        }
::= {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) ac(1)
cs1-ip-to-scp(2) version1(0)};

Core-INAP-CS1-SCP-to-SSP-AC APPLICATION-CONTEXT
-- dialogue initiated by SCP with InitiateCallAttempt
RESPONDER CONSUMER OF {
        Assist-connection-establishment-ASE,
        Generic-disconnect-resource-ASE,
        Non-assisted-connection-establishment-ASE,
        Connect-ASE
        Call-handling-ASE,
        BCSM-event-handling-ASE,
        Charging-event-handling-ASE,
        SSF-call-processing-ASE,
        SCF-call-initiation-ASE,
        Timer-ASE,
        Billing-ASE,
        Charging-ASE,
        Call-report-ASE,
        Signalling-control-ASE,
        Specialized-resource-control-ASE,
        Cancel-ASE,
        Activity-test-ASE
        }
::= {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) ac(1)
cs1-scp-to-ssp(3) version1(0)};

Core-INAP-CS1-SCP-to-SSP-traffic-management-AC APPLICATION-CONTEXT
-- dialogue initiated by SCP with CallGap
RESPONDER CONSUMER OF {
        traffic-management-ASE
        }
::= {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) ac(1)
cs1-scp-to-ssp-traffic-management(4) version1(0)};

Core-INAP-CS1-SCP-to-SSP-service-management-AC APPLICATION-CONTEXT
-- dialogue initiated by SCP with ActivateServiceFiltering
RESPONDER CONSUMER OF {
        service-management-ASE
        }
::= {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) ac(1)
cs1-scp-to-ssp-service-management(5) version1(0)};

Core-INAP-CS1-SSP-to-SCP-service-management-AC APPLICATION-CONTEXT
-- dialogue initiated by SSP with ServiceFilteringResponse
INITIATOR CONSUMER OF {
        service-management-ASE
        }
::= {ccitt(0) identified-organization(4) etsi(0) inDomain(1) in-network(1) ac(1)
cs1-ssp-to-scp-service-management(6) version1(0)};
```

7 Application entity procedures

7.1 SSF application entity procedures

7.1.1 General

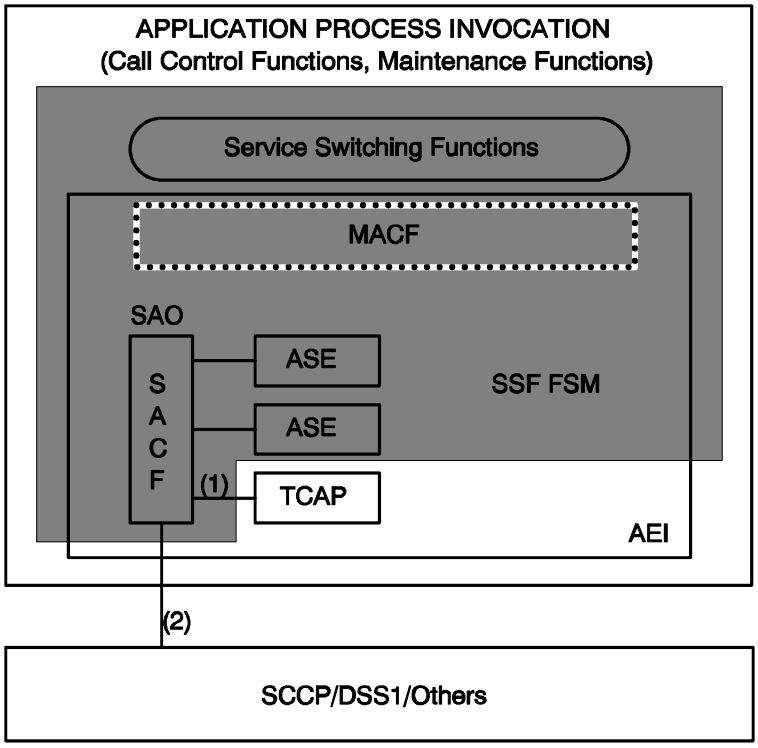
This subclause provides the definition of the SSF AE procedures related to the SSP-SCP interface. The procedures are based on the use of Signalling System No.7; other signalling systems may be used (e.g. DSS1, layer 3). Capabilities not explicitly covered by these procedures may be supported in an implementation dependent manner in the SSP, while remaining in line with Clause 6 of this ETS.

The AE, following the architecture defined in ITU-T Recommendations Q.700 [9] and Q.1400 [13], and ETS 300 287 [5] (ITU-T Recommendation Q.771), includes TCAP and one or more ASEs called TC-users. The following subclauses define the TC-user ASE which interfaces with TCAP using the primitives specified in ETS 300 287 [5] (ITU-T Recommendation Q.771); other signalling systems, such as DSS1 (layer 3), may be used.

The procedure may equally be used with other signalling message transport systems supporting the Application Layer structures defined. In case interpretations for the AE procedures defined in the following differ from detailed procedures and the rules for using of TCAP service, the statements and rules contained in the detailed Clauses 9 and 10 shall be followed.

7.1.2 Model and interfaces

The functional model of the AE-SSF is shown in figure 9; the ASEs interface to TCAP to communicate with the SCF, and interface to the Call Control Function (CCF) and the maintenance functions already defined for switching systems. The scope of this ETS is limited to the shaded area in figure 9.



(1) TC-Primitives
(2) N-Primitives

NOTE: The SSF Finite State Model (FSM) includes several Finite State Machines.

Figure 9: Functional model of SSF AE

The interfaces shown in figure 9 use the TC-user ASE primitives specified in ETS 300 287 [5] (ITU-T Recommendation Q.771) (interface (1)) and N-Primitives specified in ETS 300 009 [2] (ITU-T Recommendation Q.711) (interface (2)). The operations and parameters of INAP are defined in Clause 6 of this ETS.

7.1.3 Relations between SSF FSM and the CCF and maintenance functions

The primitive interface between the SSF FSM and the CCF/maintenance functions is an internal interface and is not subject to standardization in CS1. Nevertheless, this interface should be in line with the Basic Call State Model (BCSM) defined in ITU-T Recommendation Q.1214 [11], § 4.2.1.2.

The relationship between the BCSM and the SSF FSM may be described as follows for the case of a call/attempt initiated by an end user, and the case of a call/attempt initiated by IN Service Logic (SL):

- when a call/attempt is initiated by an end user and processed at an exchange, an instance of a BCSM is created. As the BCSM proceeds, it encounters Detection Points (DPs, see ITU-T Recommendation Q.1214 [11], § 4.2). If a DP is armed as a Trigger Detection Point (TDP), an instance of an SSF FSM is created;

- if an initiateCallAttempt is received from the SCF, an instance of a BCSM is created, as well as an instance of an SSF FSM.

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 SSME-control interfaces the different SSF FSMs and SSME FSMs respectively and the Functional Entity Access Manager (FEAM). Figure 10 shows the SSF interfaces.

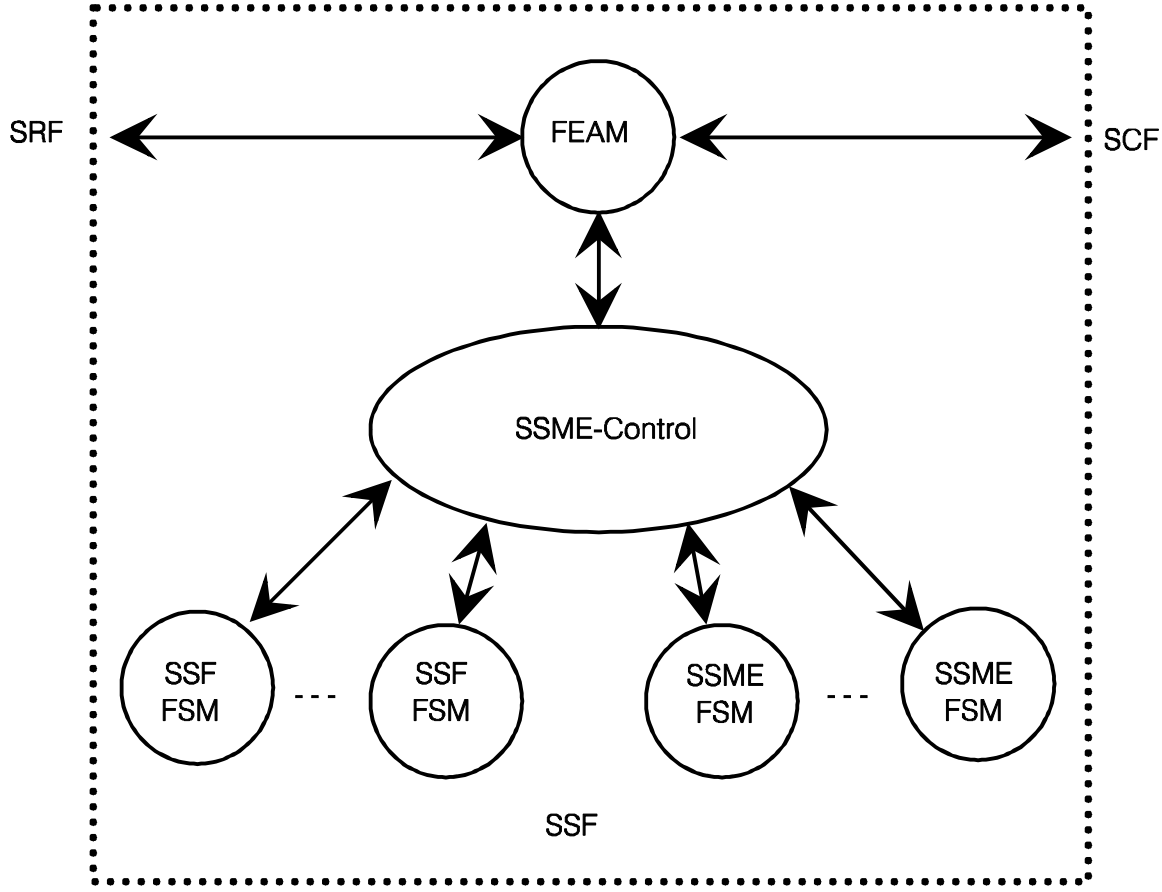


Figure 10: SSF interfaces

The FEAM provides the low level interface maintenance functions including the following:

- establishing and maintaining the interfaces to the SCF and SRF;

- passing and queueing (when necessary) the messages received from the SCF and SRF to the SSME-Control;

- formatting, queueing (when necessary), and sending the messages received from the SSME-Control to the SCF and SRF.

The SSME-control maintains the dialogues with the SCF, and SRF on behalf of all instances of the SSF FSM. These instances of the SSF FSM occur concurrently and asynchronously as calls occur, which explains the need for a single entity that performs the task of creation, invocation, and maintenance of the SSF FSMs. In particular the SSME-control performs the following tasks:

- interprets the input messages from other FEs and translates them into corresponding SSF FSM events;

- translates the SSF FSM outputs into corresponding messages to other FEs;

- captures asynchronous (with call processing) activities related to management or supervisory functions in the SSF and creates an instance of a SSME FSM. For example, the SSME provides non-call associated treatment due to changes in Service Filtering or Call Gapping. Therefore, the SSME-control separates the SSF FSM from the Call Gapping and Service Filtering functions by creating instances of SSME FSMs for each context of management related operations.

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 "filteringCriteria" cause the invocation of new SSME FSMs.

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 for CS1, the SSF FSM only applies to a functionally separate call portion (e.g., the originating BCSM or the terminating BCSM in a two-party call, but not both).

7.1.4 SSF Management Entity (SSME) FSM

The SSME FSM State Diagram is described in figure 11. The SSME FSM is independent of the individual SSF FSMs.

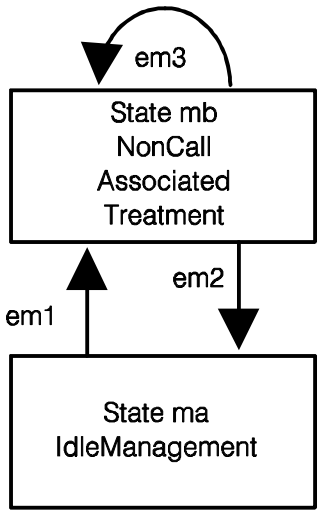


Figure 11: SSME FSM state diagram

The **Non-Call Associated Treatment** state is entered from the **IdleManagement** state when one of the following non-call associated operations is received (transition em1):

- **ActivateServiceFiltering;**
- **CallGap;**
- **ActivityTest.**

The CallGap operation may be received inside as well as outside a call context transaction. The ActivityTest operation applies to call associated transactions only. The ActivateServiceFiltering operation may be received outside a call context only.

During this state the following events can occur:

- given that Service Filtering is active, the SSF needs to send a Service Filtering Response to the SCF: the SSME FSM remains in this state (transition em3);

- given that Service Filtering is active and the Service Filtering duration expires: the SSME FSM should move to the **IdleManagement** state (transition em2) and send a ServiceFilteringResponse operation to the SCF;

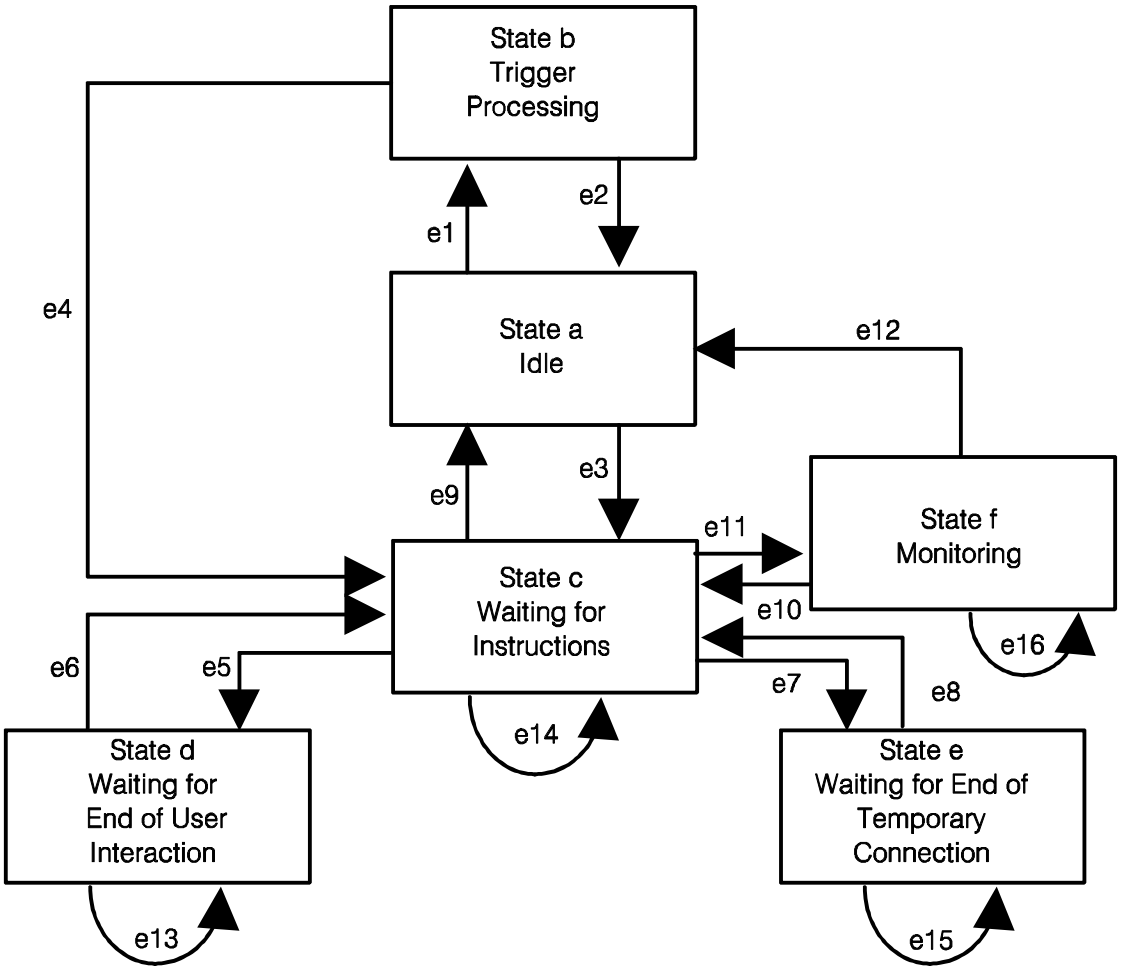
- if Call Gap related duration timer expires, the SSME FSM should move to the **IdleManagement** state (transition em2);

- given that Call Gap/Service Filtering is active, another CallGap/ActivateServiceFiltering operation could be received by the SSF, which has the same gapping/filtering criteria: the second "filter" or "gap" replace the first one (transition em3) unless the duration timer value is equal to zero, in which case the SSF should move 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 SSF FSM.

7.1.5 SSF state transition diagram

Figure 12 shows the state diagram of the SSF part of the SSP during the processing of an IN call/attempt.



NOTE: Abandon and Disconnect transitions are not shown.

Figure 12: SSF FSM state diagram

Each state is discussed in the following subclauses. General rules applicable to more than one 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;

- the SSF examines subsequent operations in the sequence. As long as sequential execution of these operations would leave the FSM in the same state, it will execute them (e.g., RequestReportBCSMEvent). If a subsequent operation causes a transition out of the state then the following operations should be buffered until the current operation has been executed. In all other cases, await an event that would cause a transition out of the current state (such an event would be the completion of operation being executed), or reception of an external event.

EXAMPLE: The SSF receives the operations FurnishChargingInformation, ConnectToResource, and PlayAnnouncement in a component sequence inside a single TCAP message. Upon receipt of this message, these operations are executed up to and including ConnectToResource while the SSF is in the **Waiting for Instruction** state. As the ConnectToResource operation is executed (and when, or after the FurnishChargingInformation operation has been completed), the SSF FSM will transition 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.

- if there is an error in processing one of the operations in the sequence, the SSF FSM processes the error (see below) and discards all remaining operations in the sequence;

- if an operation is not understood or is out of context (i.e. violates the SACF rules defined by the SSF FSM) as described above, the SSF FSM processes the error according to the rules given in subclause 10.2 (using TC-U-REJECT or the operation error UnexpectedComponentSequence).

In any state, if there is an error in a received operation, the maintenance functions are informed. Generally, the SSF FSM remains in the same state as when it received the erroneous operation, however different error treatment is possible in specific cases as described in Clause 8; depending on the class of the operation, the error could be reported by the SSF to the SCF using the appropriate component (see ETS 300 287 [5] (ITU-T Recommendation Q.774)).

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 SSF FSM should instruct the CCF 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 Abandon DP is not armed and there is no CallInformationRequest pending, then transition to the **Idle** state;

- if the Abandon DP is not armed and there is a CallInformationRequestpending, send a CallInformationReport, then transition to the **Idle** state;

- if the Abandon DP is armed as an EDP-R, send an EventReportBCSM operation, then transition to the **Waiting for Instructions** state (whether or not there is a pending CallInformationRequest);

- if the Abandon DP is armed as an EDP-N and there is no CallInformationRequest pending, send an EventReportBCSM , then transition to the **Idle** state;

- if the Abandon DP is armed as an EDP-N and there is a CallInformationRequest pending, send an EventReportBCSM and a CallInformationReport, then transition to the **Idle** state.

Other pending requests that are treated in the same way as the CallInformationRequest operation in the above list is the ApplyCharging operation when the "sendCalculationToSCPIndication" parameter is set to TRUE.

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 SSF FSM should process this event as follows:

- if the Disconnect DP is not armed for that specific leg and there is no CallInformationRequest pending, transition to the **Idle** state;

- if the Disconnect DP is not armed and there is a CallInformationRequest pending, send a CallInformationReport and transition to the **Idle** state;

- if the Disconnect DP is armed as an EDP-R for that specific leg, send either an EventReportBCSM and then transition to the **Waiting for Instructions** state or if a CallInformationRequest is pending, immediately after the CallInformationReport the corresponding EventReportBCSM has to be sent;

- if the Disconnect DP is armed as an EDP-N and there is no CallInformationRequest pending, send an EventReportBCSM , then transition to the **Idle** state;

- if the Disconnect DP is armed as an EDP-N and there is a CallInformationRequest pending, send an EventReportBCSM and a CallInformationReport, then transition to the **Idle** state.

Other pending requests that are treated in the same way as the CallInformationRequest operation in the above list is the ApplyCharging operation when the "sendCalculationToSCPIndication" parameter is set to TRUE.

The SSF 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.

Timer T_{SSF} is set in the following cases:

- when the SSF sends an InitialDP (refer to subclause 7.1.5.3 State c: "Waiting for Instructions") or AssistRequestInstructions operation (refer to subclause 7.1.6.2 State b: "Waiting for Instructions" in the assisting/hand-off case). While waiting for the first response from the SCF, the timer T_{SSF} can be reset only once by a ResetTimer operation. Subsequent to the first response, the timer can be reset any number of times;

- when the SSF enters the "Waiting for instructions" state (refer to subclause 7.1.5.3) under any other condition as the one listed in the previous case. In this case the SCF may reset 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 (refer to subclauses 7.1.5.4 and 7.1.5.5). In these cases the SCF may reset T_{SSF} using the ResetTimer operation any number of times.

In the three above cases, T_{SSF} may respectively have three different values as defined by the application.

When receiving or sending any operation which is different from the above, the SSF should reset T_{SSF} to the last used set value. This value is either one associated to the three different cases as listed above, or received in a ResetTimer operation, whatever occurred last. In the "Monitoring" state (refer to subclause 7.1.5.6) T_{SSF} is not used.

On expiration of T_{SSF} the SSF FSM transitions to the Idle state, aborts the interaction with the SCF and the CCF progresses the BCSM if possible.

The SSF state diagram contains the following transitions (events):

e1: TDP encountered
e2: Trigger fail
e3: Initiate call received
e4: Trigger detected
e5: User Interaction requested
e6: User Interaction ended
e7: Temporary connection created
e8: Temporary connection ended
e9: Idle return from Wait for Instruction
e10: EDP_R encountered
e11: Routing instruction received
e12: EDP_N last encountered
e13: Waiting for End of User Interaction state no change

e14: Waiting for Instruction state no change
e15: Waiting for End of Temporary Connection state no change
e16: Monitoring state no change
e17: Abandon (from any state) (not shown in the SSF state diagram)
e18: Disconnect (from any state) (not shown in the SSF state diagram)
e19: Non-call associated treatment from any state (not shown in the SSF state diagram)

The SSF state diagram contains the following states:

State a: Idle
State b: Trigger Processing
State c: Waiting for Instructions
State d: Waiting for End of User Interaction
State e: Waiting for End of Temporary Connection
State f: Monitoring

7.1.5.1 State a: "Idle"

The SSF FSM enters the **Idle** state under a variety of conditions, as described below.

The SSF FSM enters the **Idle** state when sending or receiving an ABORT TCAP primitive due to abnormal conditions in any state.

The SSF FSM enters the **Idle** state when DP processing fails in the **Trigger Processing** state (transition e2).

The SSF FSM 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 subclause 7.1.5;

- when a Connect, or ProceedCallProcessing operation is processed in the **Waiting for Instructions** state, and no EDPs are armed and there are no outstanding report requests (transition e9);

- when the application timer T_{SSF} expires in one of the states: **Waiting for Instructions**, **Waiting for End of User Interaction** or **Waiting for End of Temporary Connection** (transition e9);

- when a ReleaseCall operation is processed in **Waiting for Instructions** (transition e9) or **Monitoring** (transition e12);

- when the last EDP-N is encountered in the **Monitoring** state, and there are no EDP-Rs armed and no monitoring is active for charging events (transition e12);

- when the last charging event is encountered in the **Monitoring** state, and there are no EDPs armed (transition e12).

When transitioning to the **Idle** state, if there is a CallInformationRequest pending (see subclause 7.1.5), the SSF sends a CallInformationReport operation to the SCF before returning to **Idle**.

During this state the following call-associated events can occur:

- indication from the CCF that an armed TDP is encountered related to a possible IN call/service attempt: in this case the SSF FSM moves to the state **Trigger Processing** (transition e1);

- a message related to a new transaction containing an InitiateCallAttempt operation is received from the SCF: in this case the SSF FSM moves to the state **Waiting for Instructions** (transition e3).

Any other operation received from the SCF while the 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 ETS 300 287 [5] (ITU-T Recommendation Q.774)).

7.1.5.2 State b: "Trigger Processing"

Following a trigger detection related to an armed TDP in the BCSM, the SSF FSM is activated and moves from the **Idle** state to the **Trigger Processing** state (transition e1).

In this state, the SSF/CCF should:

- perform the DP processing actions specified in ITU-T Recommendation Q.1214 [11], § 4.2.2.5:
    - check if call gapping or service filtering mechanisms are active;
    - check for SCF accessibility;
    - determine if DP criteria are met;
    - handle service feature interactions;
- collect and verify necessary parameters for sending a InitialDetectionPoint to the SCF:
    - if successful and the DP is a TDP-R, send a generic InitialDetectionPoint to the SCF, as determined from DP processing, and transition to the **Waiting for Instructions** state (transition e4);
    - if DP processing fails, return to the **Idle** state (transition e2). DP processing fails in the following cases:
        - if CallGapping is in effect: the SSF FSM will instruct the CCF to terminate the call with the appropriate treatment;
        - if Service Filtering is in effect: the call is counted (if required) and the SSF FSM instructs the CCF to terminate the call with the appropriate treatment;
        - if a trigger criteria match is not found: the SSF FSM returns call control to the CCF;
        - if the Call is abandoned: the SSF returns call control to the CCF and continues processing as described in subclause 7.1.5;
        - if there is insufficient information to proceed (e.g. applies to open numbering plans): the SSF FSM instructs the CCF to terminate the call with a standard terminating treatment;
        - if the destination SCF is not accessible: the SSF FSM will instruct the CCF to route the call if possible (e.g. default routing to a terminating announcement);
        - if there is an existing control relationship for the call: the SSF returns call control to the CCF.

7.1.5.3 State c: "Waiting for Instructions"

This state is entered from either the **Trigger Processing** state, as indicated above (transition e4), or directly from the **Idle** state on receipt at the SSF of a TC_Begin indication primitive containing an InitiateCall operation from the SCF (transition e3), or from the state **Monitoring** on detection of an EDP-R (transition e10). In the second case (transition e3), the SSF moves to the **Waiting for Instructions** state immediately and then proceeds according to the contents of the InitiateCall operation and any other received operations.

In this state the SSF FSM 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 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 subclause 7.1.5;

- the application Timer T_{SSF} expires: the SSF FSM 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;

- an operation is received from the SCF: The SSF FSM acts according to the operation received as 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 e14):

- **RequestReportBCSMEvent;**
- **RequestNotificationChargingEvent;**
- **ResetTimer;**
- **FurnishChargingInformation;**
- **ApplyCharging;**
- **CallInformationRequest;**
- **SendChargingInformation;**
- **Cancel.**

The following operations may be received from the SCF and processed by the SSF, causing a state transition to **Waiting for End of User Interaction** state (transition e5):

- **ConnectToResource.**

The following operations may be received from the SCF and processed by the SSF, causing a state transition to **Waiting for End of Temporary Connection** state (transition e7):

- **EstablishTemporaryConnection.**

The following operations may be received from the SCF and processed by the SSF, causing a state transition to either **Monitoring** state (if any EDPs were armed or any reports were requested) (transition e11) or **Idle** state (transition e9):

- **Connect;**
- **CollectInformation;**
- **Continue;**
- **ReleaseCall.**

ReleaseCall operation may be received from SCF and processed by the SSF, causing a transition to **Idle** state (transition e9). If there is a CallInformationRequest, the SSF sends a **CallInformationReport** operation to the SCF before returning to **Idle**.

InitiateCallAttempt operation, if received in **Idle** state, should be processed in **Waiting for Instructions** state (transition e3).

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 subclause 7.1.5.

7.1.5.4 State d: "Waiting for End of User Interaction"

The SSF enters this state from the **Waiting for Instructions** state (transition e5) on the reception of one of the following operations:

ConnectToResource

During this state the following events can occur:

- a valid SCF-SRF operation (i.e., PlayAnnouncement, PromptAndCollectUserInformation and CancelAnnouncement) for relaying is received and is correct, the operation is transferred to the SRF for execution. The SSF FSM remains in the **Waiting for End of User Interaction** state (transition e13);

- a valid SRF-SCF operation (i.e., SpecializedResourceReport and return result from PromptAndCollectUserInformation) for relaying is received and is correct, the operation is transferred to the SCF. The SSF FSM remains in the **Waiting for End of User Interaction** state (transition e13);

- the application timer T_{SSF} expires: the SSF FSM 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;

- an operation is received from the SCF: The SSF FSM acts according to the operation received as described below;

- the user abandons. This should be processed in accordance with the general rules in subclause 7.1.5.

The following operations may be received from the SCF and processed by the SSF with no resulting transition to a different state (transition e13):

- **RequestNotificationChargingEvent;**
- **ResetTimer;**
- **FurnishChargingInformation;**
- **ApplyCharging;**
- **SendChargingInformation.**

The **DisconnectForwardConnection** 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 subclause 7.1.5.

7.1.5.5 State e: "Waiting for End of Temporary Connection"

The SSF enters this state from the **Waiting for Instructions** state (transition e7) upon receiving an EstablishTemporaryConnection operation.

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.

During this state the following events can occur:

- the application Timer T_{SSF} expires: the SSF FSM 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;

- the receipt of an indication of disconnection of forward connection from the CCF. In this case, the SSF moves to the **Waiting for Instructions** state (transition e8). The disconnection is not transferred to the Calling party;

- the user abandons. This should be processed in accordance with the general rules in subclause 7.1.5;

- an operation is received from the SCF. The 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 e15):

- **RequestNotificationChargingEvent;**
- **ResetTimer;**
- **FurnishChargingInformation;**
- **ApplyCharging;**
- **SendChargingInformation.**

The **DisconnectForwardConnection** 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 subclause 7.1.5.

7.1.5.6 State f: "Monitoring"

The SSF enters this state from the **Waiting for Instructions** state (transition e11) upon receiving a Connect, CollectInformation, Continue operation when one or more EDPs are armed or/and there is CallInformationRequest pending (see subclause 7.1.5).

In this state the timer T_{SSF} is not used; i.e., the expiration of T_{SSF} does not have any impact on the SSF FSM.

During this state the following events can occur:

- an EDP-N should be reported to the SCF by sending an EventReportBCSM operation; the SSF FSM should remain in the **Monitoring** state (transition e16) if one or more EDPs are armed or there are report requests pending. The SSF FSM shall move to the **Idle** state (transition e12) if there are no remaining EDPs armed and there are no report requests pending;

- an EDP-R should be reported to the SCF by sending an EventReportBCSM operation; the SSF FSM should move to the **Waiting for Instructions** state (transition e10);

- 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 SSF FSM transitions to the **Idle** state (transition e12), disassociating the SSF FSM from the call;

- an operation is received from the SCF: The SSF FSM 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 subclause 7.1.5.

The following operations can be received from the SCF and processed by the SSF with no resulting transition to a different state (transition e16):

- **RequestReportBCSMEvent;**
- **RequestNotificationChargingEvent;**
- **SendChargingInformation.**

The Cancel operation may be received from the SCF and processed by the SSF, causing a state transition to the **Idle** state (transition e12).

The ReleaseCall operation may be received from the SCF and processed by the SSF, causing a state transition to the **Idle** state (transition e12). If there is a CallInformationRequest pending, the SSF sends a CallInformationReport to the SCF.

Any other operation received in this state should be processed in accordance with the general rules in subclause 7.1.5.

7.1.6 Assisting/hand-off SSF FSM

The present subclause describes the SSF FSM related to the Assisting/hand-off SSF. The Assisting SSF is structured as defined in subclauses 7.1.1 through 7.1.5 of this ETS. The Hand-off FSM for CS1 applies only to the case where final treatment is to be applied. Within this subclause, the term "Assisting SSF" refers to both Assisting and Hand-off SSFs unless an explicit indication to one or the other is provided.

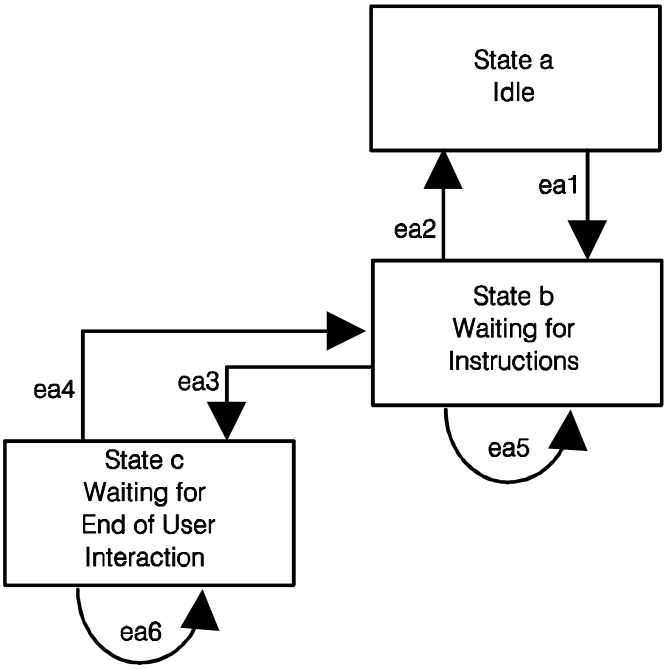


Figure 13: Assisting/hand-off SSF FSM state diagram

The Assisting/hand-off SSF state diagram contains the following transitions (events):

ea1: Assist/hand-off detected
ea2: Assist/hand-off ended (fail or success)
ea3: User Interaction requested
ea4: User Interaction ended
ea5: Waiting for Instruction state no change
ea6: Waiting for End of User Interaction state no change

The Assisting/hand-off SSF state diagram contains the following states:

State a: Idle
State b: Waiting for Instructions
State c: Waiting for End of User Interaction

7.1.6.1 State a: "Idle"

The SSF FSM 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 upstream SSF and the Assisting SSF, when a bearer channel disconnect is received from the upstream 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 Assisting SSF FSM transitions 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 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 ETS 300 287 [5] (ITU-T Recommendation Q.774).

7.1.6.2 State b: "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 assist is required, based on an implementation dependent detection mechanism(transition ea1).

In this state the SSF sends an AssistRequestInstructions operation to the SCF and the Assisting SSF FSM 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 Assisting SSF FSM 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 SSF FSM acts according to the operation received as described below;

- a bearer channel disconnect is received and the FSM moves to the **Idle** state (transition ea2).

The following operations may be received from the SCF and processed by the Assisting SSF with no resulting transition to a different state (transition ea5):

- **ResetTimer;**
- **FurnishChargingInformation;**
- **ApplyCharging;**
- **SendChargingInformation.**

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.**

The following operations can be received from the SCF and processed by the Hand-off SSF, causing a state transition to **Idle** state (transition ea2):

- **ReleaseCall.**

The above operation is allowed only in the Hand-off SSF. In the case where an implementation is not capable of differentiating between a Hand-off or an assisting SSF case, it may execute the ReleaseCall operation in the assisting SSF.

Any other operation received in this state should be processed in accordance with the general rules in subclause 7.1.5.

NOTE: Multiple Hand-off procedures are not covered by this ETS.

7.1.6.3 State c: "Waiting for End of User Interaction"

The Assisting SSF enters this state from the **Waiting for Instructions** state (transition ea3) on the reception of one of the following operations:

- **ConnectToResource.**

During this state the following events can occur:

- a valid SCF-SRF operation (i.e., PlayAnnouncement, PromptAndCollectUserInformation and CancelAnnouncement) for relaying is received and is correct, the operation is transferred to the SRF for execution. The SSF FSM remains in the **Waiting for End of User Interaction** state (transition ea6);

- a valid SRF-SCF operation (i.e., SpecializedResourceReport and return result from PromptAndCollectUserInformation) for relaying is received and is correct, the operation is transferred to the SCF. The SSF FSM remains in the **Waiting for End of User Interaction** state (transition ea6);

- when the SRF indicates to the Hand-off SSF the end of user interaction by initiating disconnection the Hand-off SSF FSM returns to the **Waiting for Instructions** state (transition ea4);

- the application Timer T_{SSF} expires: the SSF FSM 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;

- an operation is received from the SCF: The SSF FSM acts according to the operation received as described below;

- a bearer channel disconnect is received from the initiating SSF. The assisting SSF FSM moves to the **Idle** state, the connection to the SRF is released and the transaction is aborted.

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** 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.

Any other operation received in this state should be processed in accordance with the general rules in subclause 7.1.5.

7.2 SCF application entity procedures

7.2.1 General

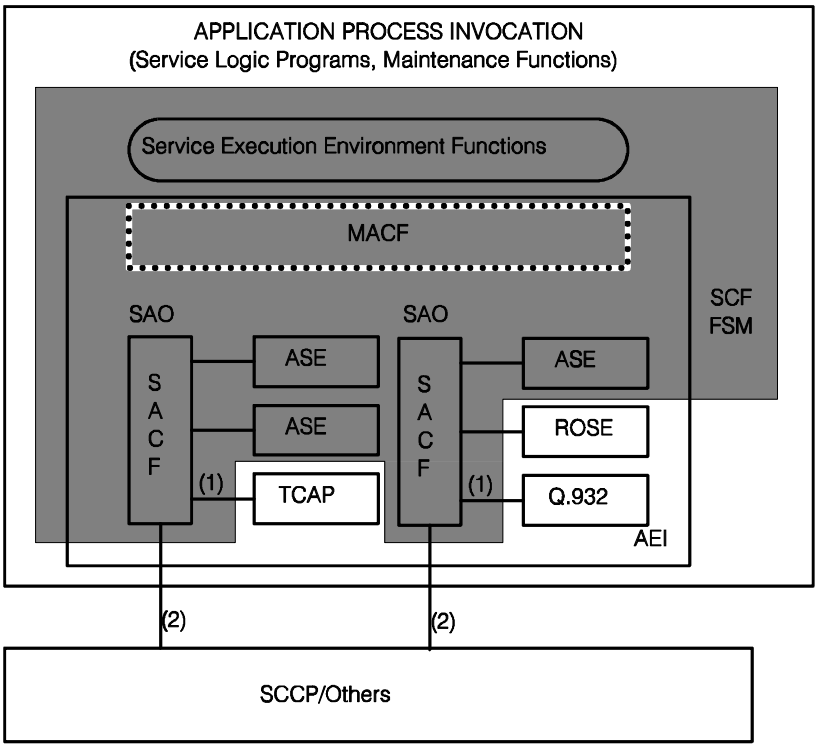
This subclause provides the definition of the SCF AE procedures related to the SCF-SSF/SRF/SDF interface. The procedures are based on the use of Signalling System No.7; other signalling systems can also be used. In addition, other capabilities may be supported in an implementation-dependent manner in the SCP, AD or SN.

The AE, following the architecture defined in ITU-T Recommendations Q.700 [9] and Q.1400 [13], and ETS 300 287 [5] (ITU-T Recommendation Q.771), includes TCAP and one or more ASEs called TC-users. The following subclauses define the TC-user ASE and SACF/MACF rules, which interface with TCAP using the primitives specified in ETS 300 287 [5] (ITU-T Recommendation Q.771).

The procedure may equally be used with other message-based signalling systems supporting the Application Layer structures defined. By no means is this text intended to dictate any limitations to Service Logic Programs (SLPs). In case interpretations for the AE procedures defined in the following differ from detailed procedures and the rules for using of TCAP service, the statements and rules contained in the detailed Clauses 9 and 10 shall be followed.

7.2.2 Model and Interfaces

The functional model of the AE-SCF is shown in figure 14; the ASEs interface with supporting protocol layers to communicate with the SSF, SRF and SDF, and interface to the SLPs and maintenance functions. The scope of this ETS is limited to the shaded area in figure 14.



(1) TC-Primitives
(2) N-Primitives

NOTE: The SCF FSM includes several Finite State Machines.

Figure 14: Functional model of SCF AE

The interfaces shown in figure 14 use the TC-user ASE primitives specified in ETS 300 287 [5] (ITU-T Recommendation Q.771) (interface (1)) and N-primitives specified in ETS 300 009 [2] (ITU-T Recommendation Q.711) (interface (2)). The operations and parameters of INAP are defined in Clause 6 of this ETS.

7.2.3 Relationship between the SCF FSM and Service Logic Programs (SLPs)/maintenance functions

The primitive interface between the SCF FSM and the SLPs/maintenance functions is an internal interface and is not a subject for standardization in CS1.

The relationship between the SLP and the SCF FSM may be described as follows (for cases both a call initiated by an end user and a call initiated by IN SL):

- 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 SL, an instance of the SCSM is created.

In either case, the SCF FSM handles the interaction with the SSF FSM (and the SRF FSM and SDF FSM) as required, and notifies the SLP of events as needed.

The management functions related to the execution of operations received from the SCF are executed by the SCF Management Entity (SCME). The SCME is comprised of the SCME-Control and multiple instances of SCME FSMs. The SCME-Control interfaces different SCSMs and the FEAM. Figure 15 shows the SCF FSM structure.

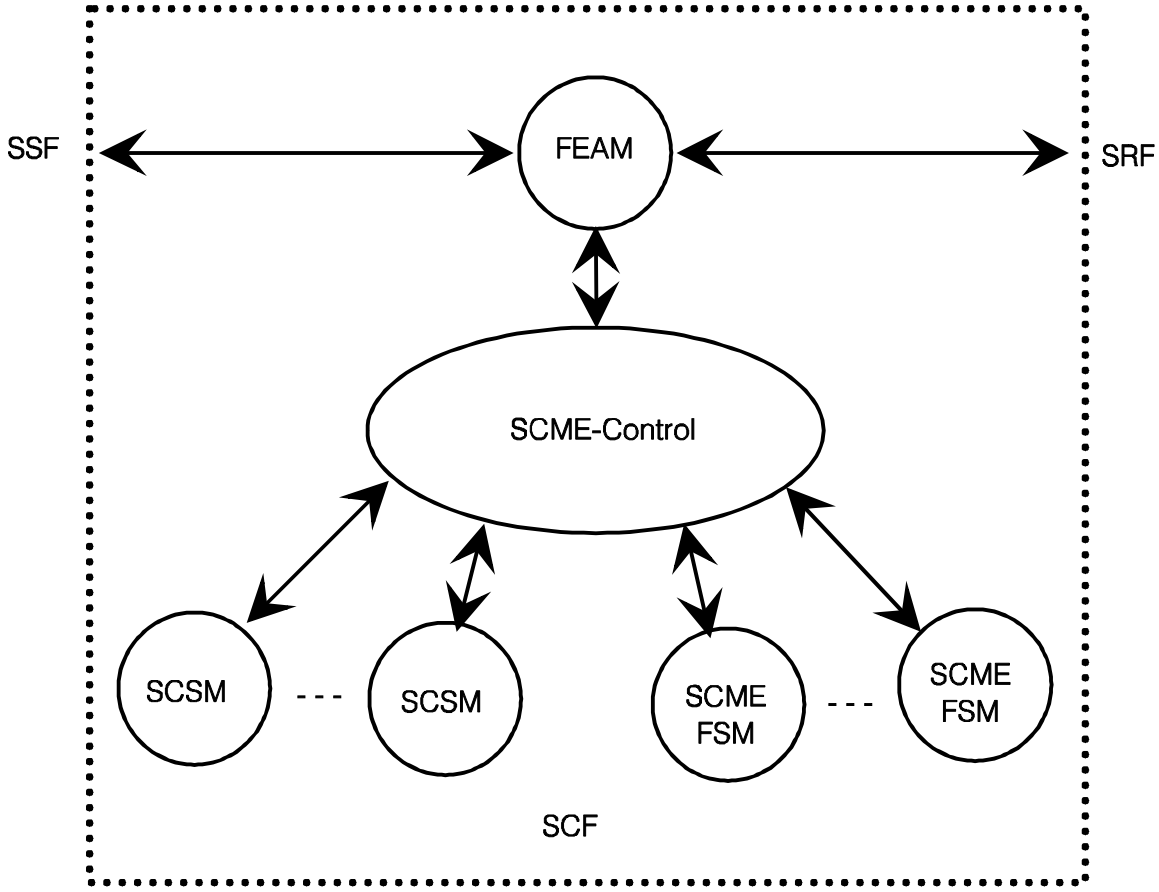


Figure 15: SCF FSM structure

The following text systematically describes the procedural aspects of the interface between the SCF and other functional entities, with the main goal of specifying the proper order of operations rather than the functional capabilities of the entities. Consequently, this text describes only a subset of the SCF functional capabilities.

The procedural model associates an SCSM with each query from the SSF. The SCSM maintains dialogues with the SSF, SRF, and SDF on behalf of SL.

Multiple requests may be executed concurrently and asynchronously by the SCF, which explains the need for a single entity that performs the tasks of creation, invocation, and maintenance of the SCF FSM objects. This entity is called the SCF Management Entity-Control (SCME-Control). In addition to the above tasks, the SCME maintains the dialogues with the SSF, SDF, and SRF on behalf of all instances of the SCF FSMs. In particular, the SCME-Control:

- interprets the input messages from other FEs and translates them into corresponding SCSM events;

- translates the SCSM outputs into corresponding messages to other FEs;

- performs some asynchronous (with call processing) activities (one such activity is flow control). It is the responsibility of the SCME-control to detect nodal overload and send the Overload Indication (e.g., Automatic Call Gap) to the SSF to place flow control on queries. Other such activities include non-call associated treatment due to changes in Service Filtering, Call Gapping, or Resource Monitoring status;

- supports persistent interactions between the SCF and other FEs;

- captures asynchronous (with call processing) activities related to management and supervisory functions in the SCF and creates an instance of a SCME FSM. For example, the SCME provides the non-call associated treatment due to changes in Service Filtering. Therefore, the SCME-Control separates the SCSM from the Service Filtering by creating instances of SCME FSMs for each context of related operations.

The different contexts of the SCME 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 SCME FSM handling this specific service filtering instance. For example, ActivateServiceFiltering operations providing different "filteringCriteria" cause the invocation of new SCME FSMs.

Finally, the FEAM relieves the SCME of low-level interface functions. The functions in the FEAM include:

- establishing and maintaining the interfaces to the SSF, SRF and SDF;

- passing (and queueing when necessary) the messages received from the SSF, SRF, and SDF to the SCME; and

- formatting, queueing (when necessary), and sending messages received from the SCME to the SSF, SRF, and SDF.

  NOTE: Although the SCSM includes a state and procedures concerning queue management, this type of resource management only represents one way of managing network call queues. Another alternative is to let the SSF/CCF manage call queues; however, the technical details of how the SSF/CCF performs queue management is beyond the scope of IN. As such, the RCO (see subclause 7.2.4.3) and the queueing state of the SCSM (State 2.2), along with its relevant sub-states, events and procedures, are only required and applicable in the case where queue management is performed in the SCF.

7.2.4 Partial SCF Management Entity (SCME) state transition diagram

The key part of the SCME state diagram is described in figure 16.

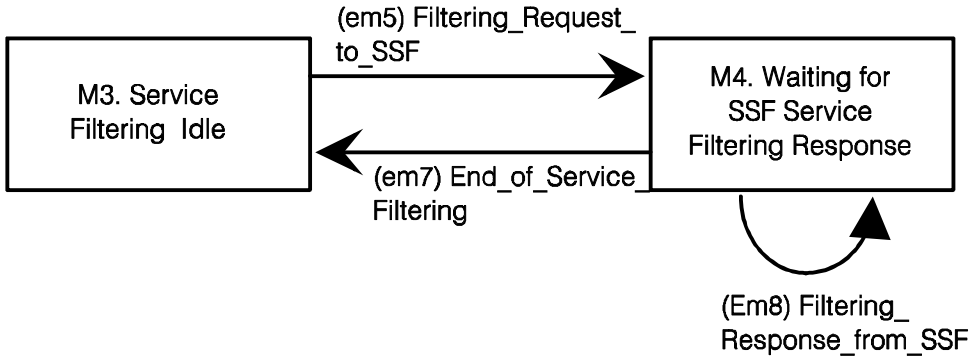


Figure 16: The service filtering FSM in the SCME

The SCME handles the following operations:

- **ActivateServiceFiltering;**
- **ServiceFilteringResponse;**
- **CallGap;**
- **ActivityTest.**

The issuing of the **CallGap** and **ActivityTest** operations does not cause state transitions in the SCME. The procedures for the rest of the above operations are described below.

The operations that are not listed above do not affect the state of the SCME; these operations are passed to the relevant SCSM.

7.2.4.1 State M3: "Service Filtering Idle"

The following event[1)] is considered in this state:

- (em5) Filtering_Request_to_SSF: This is an internal event, caused by the need of the SL 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**.

7.2.4.2 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:

- (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 **RequestServiceFiltering** operation previously issued to the SSF. This event does not cause a transition out of this state, and the SCSM remains in state M4, **"Waiting for SSF Service Filtering Response"**.

When Service Filtering is active, another **ServiceFiltering** operation could be sent to the SSF that has the same filtering criteria; this second "filter" replaces the first one.

---

1) All 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.

7.2.4.3 The Resource Control Object (RCO)

The 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 methods of the RCO; and

2) the RCO Methods.

For the purposes of this ETS, no implementation constraints are placed on the structure. The only requirement to the structure is that, for each supported resource, it:

1) stores the status of the resource (e.g., Busy or Idle); and

2) maintains the queue of SCSMs that are waiting for this resource.

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 Queueing Timer has expired or the call has been abandoned.

7.2.5 The SCF Call State Model (SCSM)

Figure 17 shows the general state diagram of the SCSM 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 subclauses. Each state, except **"Idle"**, **"SDF Request Idle"** and **"Waiting for SDF Response"**, has internal sub-FSMs composed of the sub-states.

General rules applicable to more than one state are as follows:

In every state, if there is an error in a received operation, the SLP and the maintenance functions are informed. Generally, the SCSM remains in the same state, however, different error treatment is possible in specific cases as described in Clause 8. Depending on the class of the operation, the error can be reported to the SSF, SRF, or SDF (see ETS 300 287 [5] (ITU-T Recommendation Q.774)).

It also holds that, in every state, if the SCSM is informed that the dialogue with the SSF is terminated, then it informs the SLP and returns to the Idle state. In this case, all resources allocated to that call, including those required for relevant dialogues with the other functions, shall be de-allocated. To simplify the diagram, such transitions are not demonstrated in the figures.

When the SLP requests call information, the SCSM transmits the **CallInformationRequest** operation to the SSF, and then **CallInformationReport** is outstanding.

In any state (except **"Idle"**), the SCSM may receive the **CallInformationReport** operation from the SSF, when the **CallInformationReport** is outstanding.

Other pending requests that are treated in the same way as the CallInformationRequest is the ApplyCharging when the "sendCalculationToSCPIndication" parameter is set to TRUE.

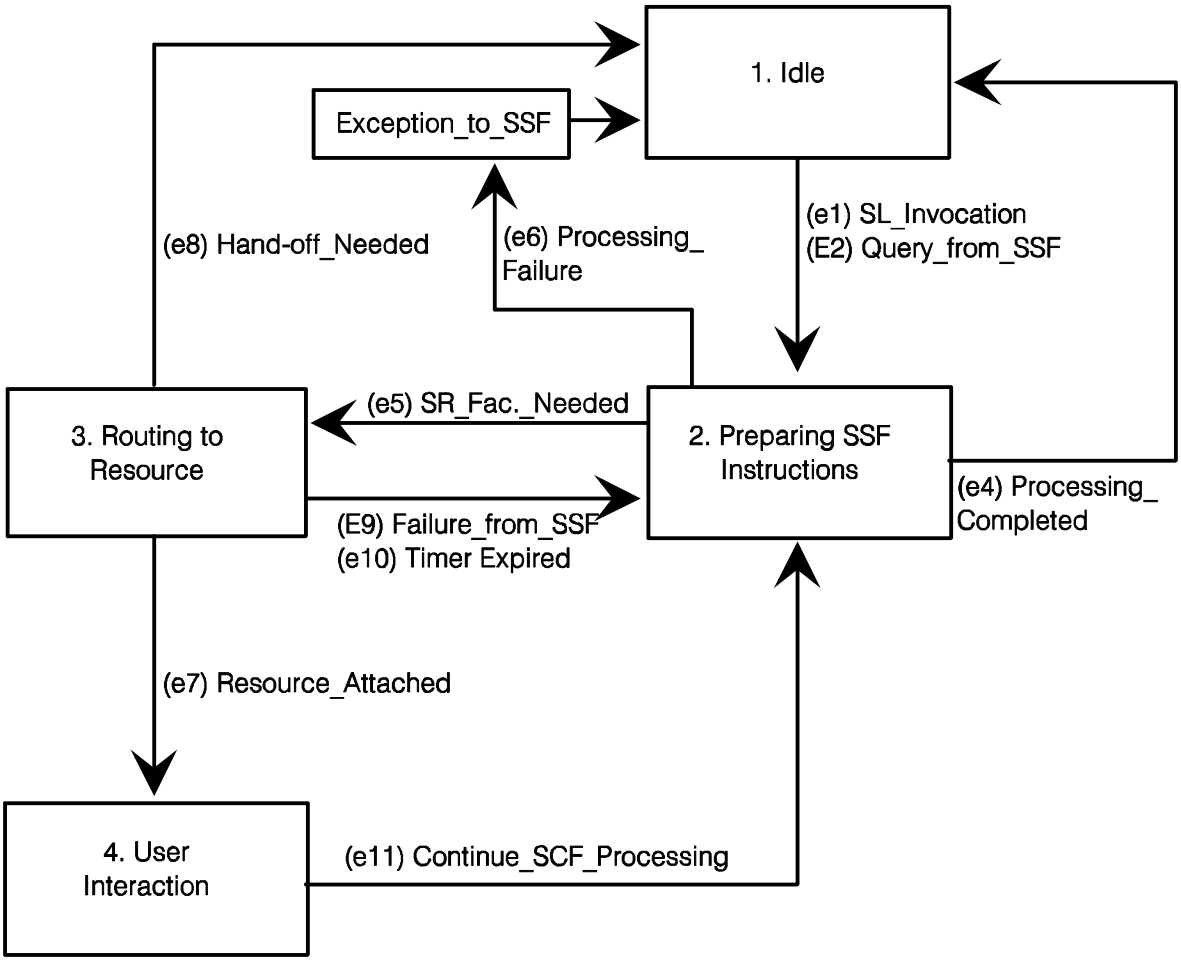


Figure 17: SCSM FSM state diagram

From any state (except **"Idle"**), if **CallInformationReport** is outstanding and the SLP indicates that the processing has been completed, the SCSM remains in the same state until it receives the **CallInformationReport** operation.

The general rules for one or a sequence of components sent in one or more TCAP messages, which may include a single operation or multiple operations, are specified in subclause 7.1.5 and are not described here.

The SCSM has an application timer, $T_{SCF-SSF}$, whose purpose is to reset the timer T_{SSF}, which is used to prevent excessive call suspension time and to guard the association between the SSF and the SCF.

Timer $T_{SCF-SSF}$ is set in the following cases:

- when the SCF receives an **InitialDP** or **AssistRequestInstructions** operation (see subclause 7.2.5.2.1, State 2.1: "**Preparing SSF Instructions**", and subclause 7.2.5.2.2.1, State 2.2.1: "**Preparing SSF Instructions**"). In this case, this timer is reset when the first request, other than **ResetTimer** operation, is sent to the SSF. On the expiration of $T_{SCF-SSF}$, the SCF may reset T_{SSF} once, using the **ResetTimer** operation, and also reset $T_{SCF-SSF}$. On second expiration of $T_{SCF-SSF}$, the SCSM informs SLP and the maintenance functions, and the SCSM transits to the **Idle** state;

- when the SCF enters the Queuing sub-state (see subclause 7.2.5.2.2, State 2.2.2: "**Queueing**"). In this case, on the expiration of timer $T_{SCF-SSF}$, the SCF may reset T_{SSF} using the **ResetTimer** operation any number of times; and

- when the SCF enters the "**Waiting for Assist Request Instructions**" state or the "**User Interaction**" state (see subclauses 7.2.5.3.2 and 7.2.5.4). In these cases, on the expiration of $T_{SCF-SSF}$, the SCF may reset T_{SSF} using the **ResetTimer** operation any number of times.

In all three cases, $T_{SCF-SSF}$ may respectively have three different values as defined by the application. The value of $T_{SCF-SSF}$ are smaller than the respective value of T_{SSF}.

When receiving or sending any other operation, the SCF should reset $T_{SCF-SSF}$. In the "**Waiting for Notification or Request**" state (see subclause 7.2.5.2.3), $T_{SCF-SSF}$ is not used.

The SCSM also has an application timer, $T_{ASSIST/HAND-OFF}$, whose purpose is to prevent excessive assist/hand-off suspension time. The SCSM sets the timer $T_{ASSIST/HAND-OFF}$ when the SCSM sends the **EstablishTemporaryConnection** or **Connect** operation with a correlation identifier (ID). This timer is stopped when the SCSM receives the **AssistRequestInstructions** operation from the assisting/handed-off SSF. On expiration of $T_{ASSIST/HAND-OFF}$, the SCSM informs SL and the maintenance functions, and the SCSM transits to the **Idle** state (in case of Hand-off) or to the **Preparing SSF instructions** state (in case of assist).

The SCSM has an additional application timer, $T_{ActTest}$, which guards a dialogue and the related resources. This timer is set at the beginning of a dialogue. The timer expiration causes a check of the dialogue and the related resources at the SSF/SCF via the operation ActivityTest (refer to the detailed procedure of that operation). In case of successful processing of that operation the timer is reset.

The call-control-related operations relevant to the SCF-SSF interface (except the SCME related operations) are categorized into:

- call-processing-related operations; and

- non-call-processing-related operations.

Call-processing-related operations are grouped into the following two sets:

- **CollectInformation;**
- **Connect;**
- **ReleaseCall;**
- **Continue,**

and

- **InitiateCallAttempt;**
- **ConnectToResource;**
- **DisconnectForwardConnection;**
- **EstablishTemporaryConnection.**

For the first set of call-processing-related 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 SL needs to send operations in parallel, they are sent in the component sequence.

In what follows, each state is described in a separate subclause together with the events that cause a transition out of this state. The outputs are presented within smaller rectangles than the states are; unlike the states and events, the outputs are not enumerated.

7.2.5.1 State 1: "Idle"

The following events are considered in this state:

- (e1) SL_Invocation: This is an internal event caused by the need of the SL to start a call. The SCSM issues the **InitiateCallAttempt** operation to the SSF;

- (E2) Query_from_SSF: This is an external event, caused by a reception of one of the following operations:

  - **InitialDP;**
  - **AssistRequestInstructions** (for the Service Hand-off case).

Both events cause a transition to State 2, **Preparing SSF Instructions**.

7.2.5.2 State 2: "Preparing SSF Instructions"

In this state, the SCF determines how to further process.

The following events are considered in this state:

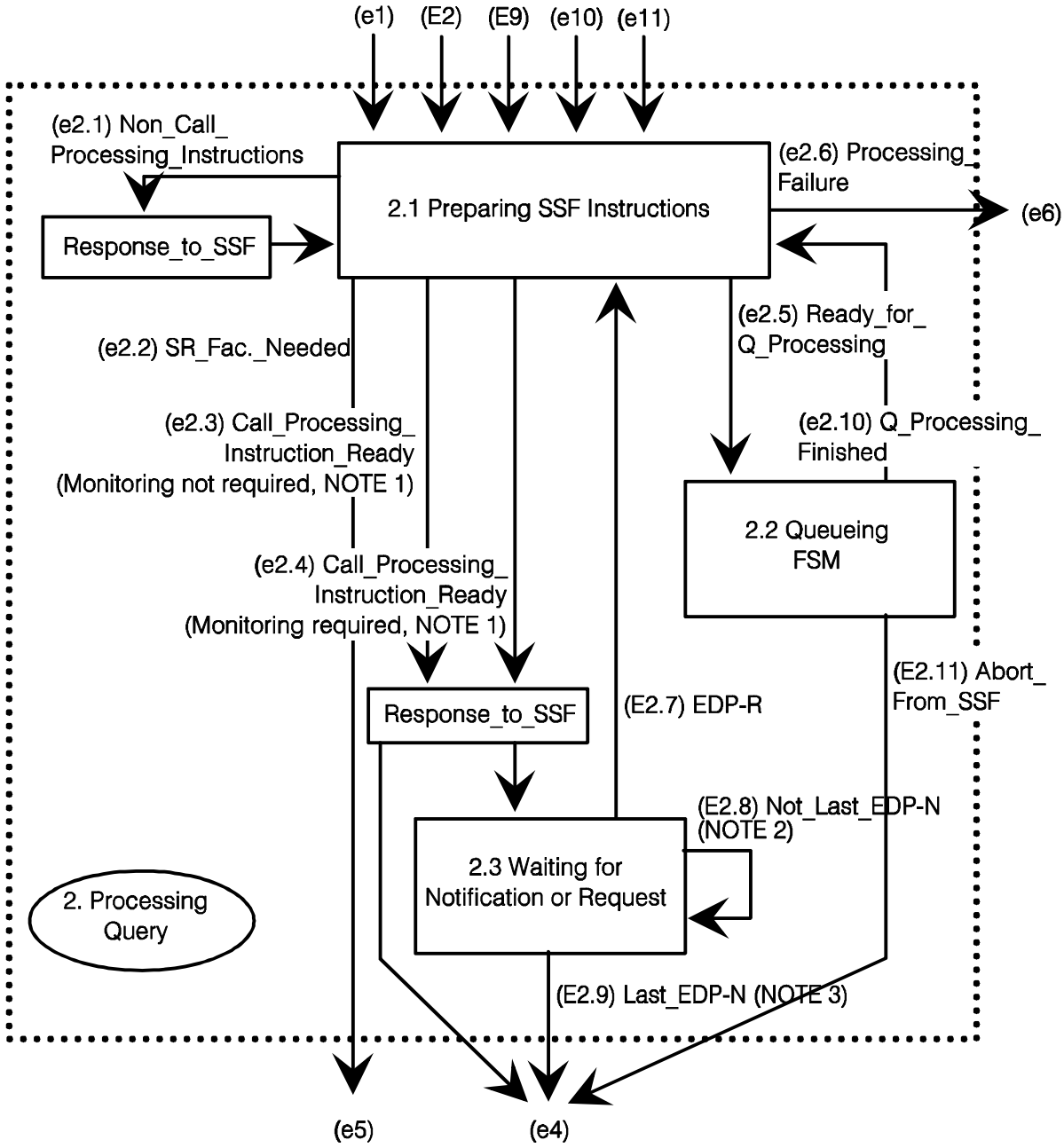
- (e4) Processing_Completed: This is an internal event. In this case, the SCF has completed the processing of the instructions to the SSF. This event causes a response to be sent to the SSF and a transition to State 1, **Idle**;

- (e5) SR_Facilities_Needed: This is an (internal) event caused by the need of the SL 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 State 3, **Routing to Resource**;

- (e6) Processing_Failure: This (internal) event causes an appropriate exception processing and a transition back to State 1, **Idle**.

  NOTE: Here and further in this ETS, the exception processing is not defined. It is assumed, however, that it has to include releasing all the involved resources and sending an appropriate response message to the SSF.

To further describe the procedures relevant to this state, the state is divided into three sub-states, which are described in the following three subclauses (this subdivision is illustrated in figure 18).



NOTE 1: Including CallInformationRequest and ApplyCharging with report request.

NOTE 2: Including CallInformationReport, ApplyChargingReport, and EventNotificationCharging.

NOTE 3: Including CallInformationReport and ApplyChargingReport.

Figure 18: Partial expansion of the State 2 FSM

7.2.5.2.1 State 2.1: "Preparing SSF Instructions"

State 2.1, **Preparing SSF Instructions**, is where the initial decision is made on whether the SDF information or a Specialized Resource is needed, whether queueing is supported, etc. In addition, the EDP-R-related processing is also performed in this state.

On entering this state, the SCSM starts or resets the timer $T_{SCF-SSF}$.

The following events are considered in this state:

- (e2.1) Non-Call_Processing_Instructions: This is an internal event caused by the SL 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;**
  - **CallInformationRequest;**
  - **FurnishChargingInformation;**
  - **RequestReportBCSMEvent;**
  - **RequestNotificationChargingEvent;**
  - **ResetTimer;**
  - **SendChargingInformation;** and
  - **Cancel (for all report requests).**

  This event causes a transition back to State 2.1, **Preparing SSF Instructions**.

- (e2.2) SR_Facilities_Needed: This is an internal event, caused by the SL when there is a need to use the SRF. This event maps into the SCSM event (e5).

- (e2.3) Call_Processing_Instruction_Ready (Monitoring not required): This is an internal event caused by the SL when the final call-processing-related operation is ready and there is no armed EDP and no outstanding **CallInformationReport** or **ApplyChargingReport** operations. It causes one of the following operations to be issued to the SSF:

  - **Connect;**
  - **Continue;**
  - **ReleaseCall.**

  In addition, one or more of the following operations may be issued to the SSF prior to the operations listed above:

  - **FurnishChargingInformation;**
  - **SendChargingInformation;**
  - **Cancel (for all report requests);** and
  - **RequestReportBCSMEvent (to disarm all armed EDPs).**

  This event maps into the SCSM event (e4).

- (e2.4) Call_Processing_Instruction_Ready (Monitoring required): This is an internal event caused by the SL when a call-processing-related operation is ready and the monitoring of the call is required (e.g., an EDP is set, or there is an outstanding **CallInformationReport** or **ApplyChargingReport**, or there is a need to issue such a request). It causes one of the following operations to be issued to the SSF:

  - **CollectInformation;**
  - **Connect;**
  - **Continue;**
  - **ReleaseCall.**

  In addition, one or more of the following operations may be issued to the SSF prior to one of the operations listed above:

  - **ApplyCharging;**
  - **CallInformationRequest;**
  - **FurnishChargingInformation;**
  - **RequestReportBCSMEvent;**
  - **RequestNotificationChargingEvent;** and
  - **SendChargingInformation.**

This event causes a transition into State 2.3, **Waiting for Notification or Request**.

- (e2.5) Ready_for_Queueing_Processing: This is an internal event caused by the SL when queueing of the call is required. This event causes a transition into State 2.2, **Queueing FSM**.

- (e2.6) Processing_Failure: This is an internal event, and it maps into the SCSM event (e6) Processing_Failure.

7.2.5.2.2 State 2.2 "Queueing FSM"

When the SCF is processing the Query from the SSF/CCF, 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.

NOTE: The manner in which the status of the resources is maintained is described in subclause 7.2.4.3.

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 operations can be sent in this state:

- **ApplyCharging;**
- **CallInformationRequest;**
- **FurnishChargingInformation;**
- **RequestReportBCSMEvent;**
- **RequestNotificationChargingEvent;**
- **ResetTimer;** and
- **SendChargingInformation.**

The following events are considered in this state:

- (e2.10) Queueing_Processing_Finished: This is an internal event caused by the SLP when it is ready to prepare the call-related operation to the SSF. This event causes a transition to State 2.1, **Preparing SSF Instructions;**

- (E2.11) Abort_from_SSF: This is an external event caused by the reception of the **Abort** message from the SSF (on call abandonment), and it causes a transition that maps into the SCSM event (e4).

This state further expands into an FSM, which is depicted in figure 19.

This FSM does not explicitly describe all possible combinations of resource monitoring functions used for queueing. The following possibilities may be used in implementations:

- monitoring based on issuing by the SCSM of the **RequestEventReportBCSM** 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, a SCF functionality may be used for scanning the status of resources.

In the rest of this subclause, the state-by-state description of the FSM is followed by the description of the supporting mechanisms of the SCME.

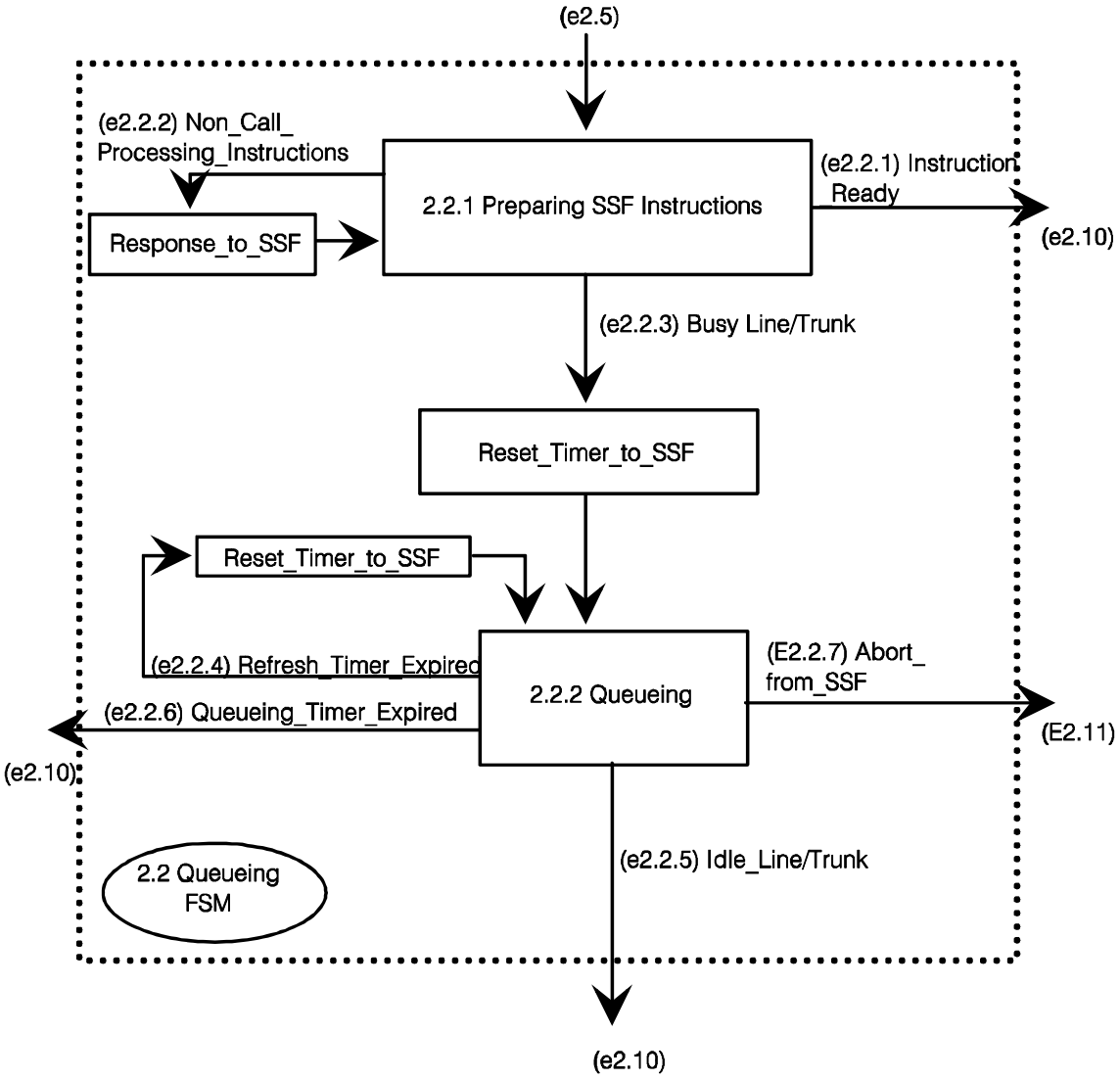


Figure 19: Partial expansion of the State 2 FSM as related to queueing

7.2.5.2.2.1 State 2.2.1: Preparing SSF Instructions

In this state, the SCSM prepares the instructions for the SSF to complete the call. The following events are considered in this state:

- (e2.2.1) Instruction_Ready: This is an internal event that takes place only when the required resource is available. In this case, the SCSM has obtained the address of the free resource via the Get_Resource method of the RCO (see subclause 7.2.4.3). This event causes a transition to the State 2.1, **Preparing SSF Instructions** (transition (e2.10)).

- (e2.2.2) Non-Call_Processing_Instructions: This is an internal event caused by the SL 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;**
  - **CallInformationRequest;**
  - **FurnishChargingInformation;**
  - **RequestReportBCSMEvent;**
  - **RequestNotificationChargingEvent;**
  - **ResetTimer;** and
  - **SendChargingInformation.**

  This event causes a transition back to State 2.2.1, **Preparing SSF Instructions**.

- (e2.2.3) Busy_Line/Trunk: This is an internal event caused by the RCO when no terminating line/trunk is available. This event causes the **ResetTimer** operation, with a suitable queueing timervalue, to be sent to the SSF, and a transition to State 2.2.2, **Queueing**.

7.2.5.2.2.2 State 2.2.2: "Queueing"

In this state, the SCSM 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 SCSM is in this state. In this ETS, the relevant further expansion of the state is not provided; it is, however, not different from that of the SCSM States 3 and 4. However, if announcements are completed before the call is dequeued and the SSF FSM has transitioned to state **Waiting for Instructions**, the ResetTimer operation should be sent to set the $T_{SCF-SSF}$ with an appropriate value. Once the SCSM enters this state, the Queueing Timer is started, and the $T_{SCF-SSF}$ is reset. The respective roles of these timers are as follows:

- the Queueing Timer limits the time that a call can spend in the queue, and its value may be customer-specific;

- the $T_{SCF-SSF}$ signals when the **ResetTimer** operation shall be sent to the SSF/CCF lest the latter abandons the call. Hence, the value of this timer is set in agreement with that of the relevant timer T_{SSF} within the SSF/CCF.

The FurnishChargingInformation operation may also be sent to the SSF at this time to indicate the initiation of queuing for call logging purposes.

The following events are considered in this state:

- (e2.2.4) Refresh_Timer_Expired: This is an internal event, which results in sending the **ResetTimer** operation to the SSF/CCF and a transition back to the same state;

- (e2.2.5) Idle_Line/Trunk: This is an internal event, which maps into the State 2 event (e2.10);

- (e2.2.6) Queueing_Timer_Expired: This is an internal event, which results in processing the Cancel method of the RCO and causes a transition out of this state to the State 2.1 **Preparing SSF Instructions** (Transition (e2.10)) (following procedures depends on decision of the SL that may play (or not play) the terminating announcement);

- (E2.2.7) Abort_from_SSF: This is an external event caused by the reception of the Abort message from the SSF (on call abandonment), and it causes a transition that maps into the State 2 event (E2.11). The SCME takes care of updating the queueing data via the Cancel method of the RCO.

7.2.5.2.3 State 2.3: "Waiting for Notification or Request"

In this state, the SCSM waits for a notification or a request from the SSF.

On entering this state the SCSM stops the timer $T_{SCF-SSF}$.

The following events are considered in this state:

- (E2.7) EDP-R: This is an external event, caused by a reception of the following operation:

  - **EventReportBCSM** (for EDP_R).

  This event causes a transition to State 2.1 **Preparing SSF Instructions**.

- (E2.8) 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 an outstanding CallInformationReport or ApplyChargingReport operation. This event causes a transition to back to State 2.3, **Waiting for Notification or Request**.

- (E2.9) Last_EDP-N: This is an external event, caused by a reception of one of the following operations:

  - **ApplyChargingReport;**
  - **CallInformationReport;**
  - **EventReportBCSM** (for EDP_N)**.**

  In this case, there is no outstanding armed EDP and no outstanding CallInformationReport or ApplyChargingReport operation. This event maps into the SCSM event (e4).

This concludes the description of State 2 **Preparing SSF Instructions**.

7.2.5.3 State 3: "Routing to Resource"

The resource is any SRF facility (e.g., Intelligent Peripheral).

In this state, interactions with the SSF are necessary. Accordingly, the following events cause transitions out of this state:

- (e7) Resource_Attached: The SRF is available. This event causes a transition to State 4, **User Interaction**;

- (e8) Hand-off_Needed: When the Hand-off procedure is initiated, the SCSM terminates the interaction with the initiating SSF. This event causes a transition to the State 1, **Idle**. When the **AssistRequestInstructions** operation from the Handed-off SSF is received by the SCME, the SCME creates the new SCSM. The SCF shall maintain sufficient information in order to correlate the subsequent AssistRequestInstructions operation (from the assisting SSF or SRF) to the existing Service Logic Program Instance (SLPI);

- (E9) Failure_from_SSF: The inability of the SSF to connect to requested resources causes a transition to State 2, **Preparing SSF Instructions**; and

- (e10) Timer_Expired: This event takes place when $T_{ASSIST/HAND-OFF}$ expires. This event causes a transition to State 2, **Preparing SSF Instructions**.

To further describe the procedures relevant to this state, the state is divided into two sub-states, which are described in the following two subclauses (this subdivision is illustrated in figure 20).

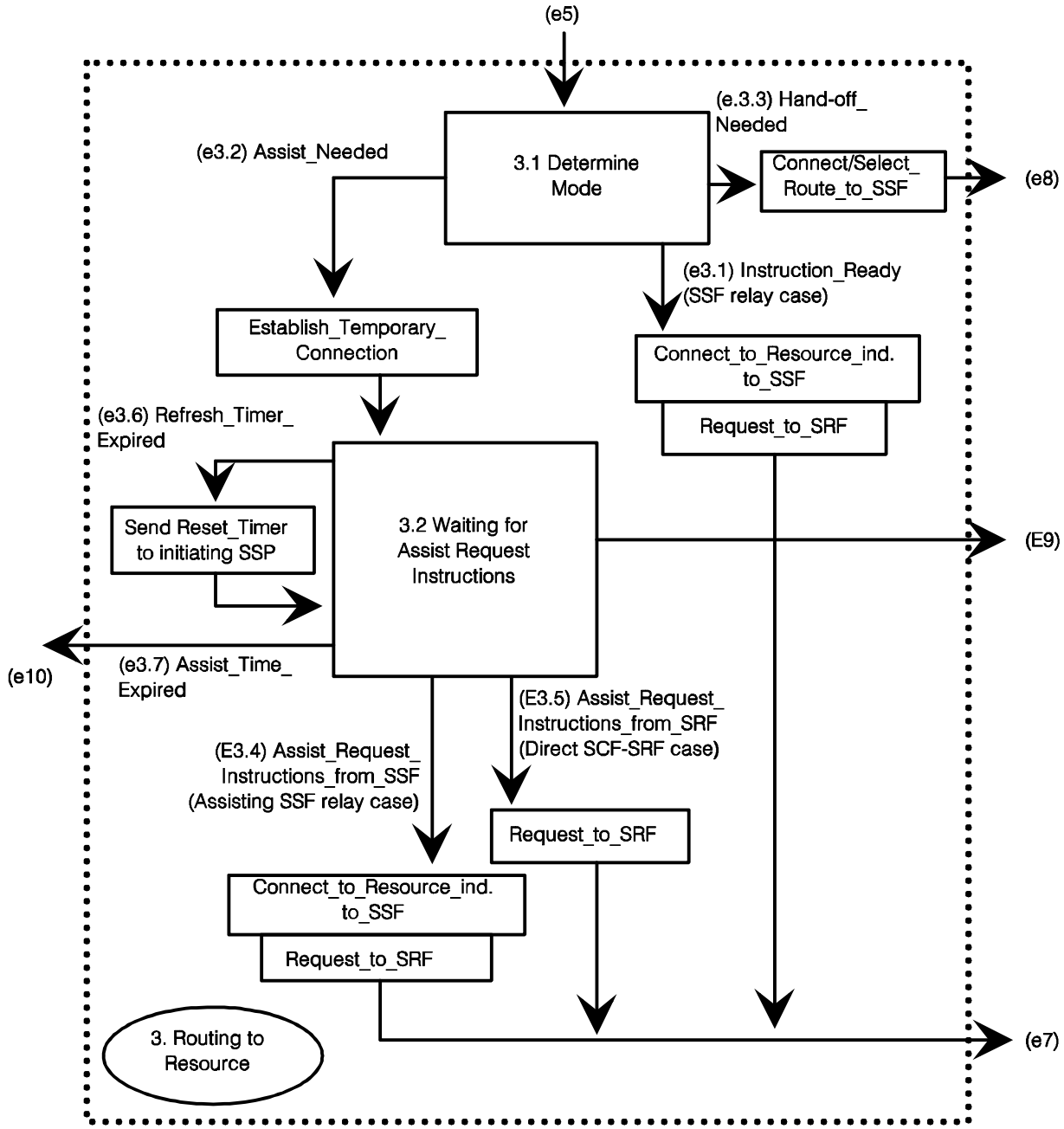


Figure 20: State 3 FSM

7.2.5.3.1 State 3.1: "Determine Mode"

In this state, the SCSM determines the User Interaction mode to connect the call to SRF. The following events are considered in this state:

- (e3.1) Instruction_Ready: This is an internal event that takes place only when the Initiating SSF relay case. In this case, the SCSM sends the **ConnectToResource** operation accompanied by **PlayAnnouncement** or **PromptAndCollectUserInformation** operation to the Initiating SSF, and transits out to the State 4 **User Interaction**. This transition maps into the event (e7);

- (e3.2) Assist_Needed: This event is an internal event that takes place when the Assisting SSF is needed or Direct SCF-SRF relation is needed. In this case, the SCSM sends the **EstablishTemporaryConnection** operation to the Initiating SSF with the Assisting SSF address or SRF address, and transits to the State 3.2 **Waiting for Assist Request Instructions**; and

- (e3.3) Hand-off_Needed: This is an internal event that takes place only when the Hand-off case. In this case, the SCSM sends the **Connect** operation with the Handed-off SSF address to the Initiating SSF, and transits to the State 1 **Idle**. This transition maps into the event (e8). The SCF shall maintain sufficient information in order to correlate the subsequent AssistRequestInstructions operation (from the assisting SSF or SRF) to the existing SLPI.

7.2.5.3.2 State 3.2: "Waiting for Assist Request Instructions"

In this state, the SCSM waits for the **AssistRequestInstructions** operation from the Assisting SSF (SSF relay case) or from the SRF (Direct SCF-SRF case). On entering this state the SCSM starts the Timer $T_{ASSIST/HAND-OFF}$, and resets the timer $T_{SCF-SSF}$. The following events are considered in this state:

- (E3.4) Assist_Request_Instructions_from_SSF (Assisting SSF relay case): This is an external event caused by the receipt of **AssistRequestInstructions** from the Assisting SSF. In this case, the SCSM transmits the **ConnectToResource** operation accompanied by **PlayAnnouncement** or **PromptAndCollectUserInformation** operation to the Assisting SSF, and transits out to the State 4, **User Interaction**. This transition maps into the event (e7);

- (E3.5) Assist_Request_Instructions_from_SRF (Direct SCF-SRF case): This is an external event caused by the receipt of **AssistRequestInstructions** from the SRF. In this case, the SCSM transmits the **PlayAnnouncement** or **PromptAndCollectUserInformation** operation to the SRF, and transits out to the State 4, **User Interaction**. This transition maps into the event (e7);

- (e3.6) Refresh_Timer_Expired: This is an internal event that takes place on the expiration of $T_{SCF-SSF}$. In this case, the SCSM transmits the **ResetTimer** operation to the Initiating SSF, and transits back to the same state;

- (e3.7) Assist_Timer_Expired: This is an internal event that takes place on the expiration of $T_{ASSIST/HAND-OFF}$. In this case, the SCSM informs the SCME and SLP, and transits to the State 2 **Preparing SSF Instructions**. This event maps into the event (e10); and

- (E3.8) Initial_SSF_Failure: This is an external event caused by the reception of SSF failure. This event causes a transition that maps into the SCSM event (E9).

7.2.5.4 State 4: "User Interaction"

In this state, the SCF requests the SRF to provide user interaction (e.g., collect additional information and/or play announcements). When an interaction is finished the SCF can instruct the SSF to disconnect the bearer between SSF and SRF. Alternatively, it can send a user interaction operation to the SRF containing an indication that allows the SRF initiated disconnect.

On entering this state the SCSM resets the timer $T_{SCF-SSF}$.

The following events cause transitions out of this state:

- (e11) 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 2, **Preparing SSF Instructions**.

To consider the processing of this state in more detail, it is expanded into a separate FSM depicted in figure 21.

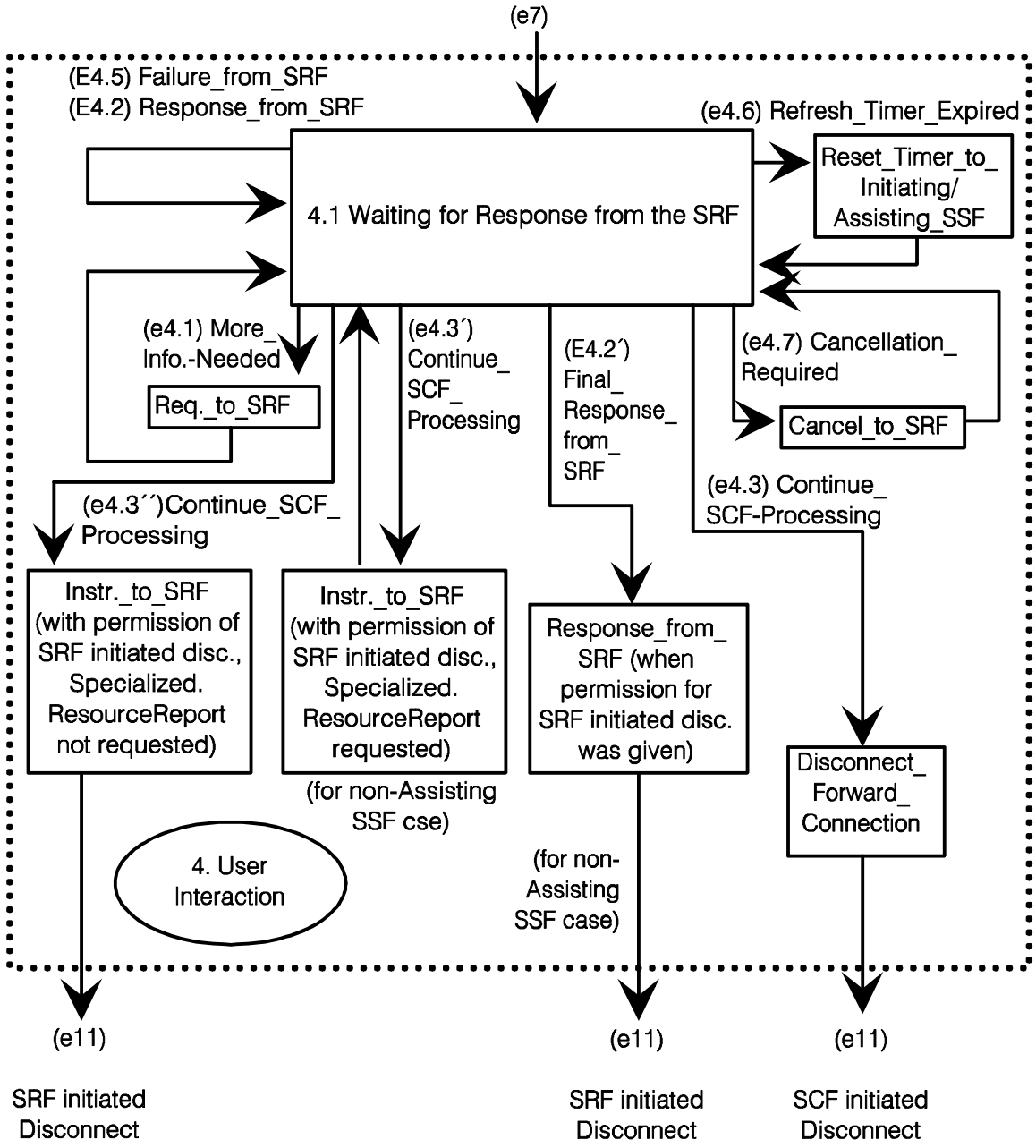


Figure 21: State 4 FSM

7.2.5.4.1 State 4.1 "Waiting for Response from the SRF"

In this state, the SCF waits for the response to the previously sent operation and evaluates this response. The following events are considered in this state:

- (e4.1) More_Information_Needed results in issuing yet another operation to the SRF; it causes a transition back to State 4.1;

- (E4.2) Response_from_SRF: This is an external event caused by the reception of **SpecializedResourceReport** or return result from **PromptAndCollectUserInformation** operation. On the receipt of this operation, the SCSM transits back to the same state;

- (E4.2') Final_Response_from_SRF: This is an external event caused by the reception of **SpecializedResourceReport** operation, as requested by the SCF, in response to the previous **PlayAnnouncement** or return result from **PromptAndCollectUserInformation** operation with permission of SRF-initiated disconnect. In the Initiating SSF relay case and the Direct SCF-SRF case, on the receipt of this event, the SCSM transits to the State 2, **Preparing SSF Instructions**. This event maps into the event (e11);

- (e4.3) Continue_SCF_Processing: This is an internal event that takes place when the SCSM 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 SCSM sends the **DisconnectForwardConnection** operation to the Initiating SSF and transits to the State 2, **Preparing SSF Instructions**. This event maps into the event (e11);

- (e4.3') Continue_SCF_Processing: This is an internal event that takes place when the SCSM finishes the User Interaction and requests the disconnection of bearer connection between the Initiating SSF and SRF by means of SRF-initiated disconnect, while an SpecializedResourceReport operation is requested to be returned to the SCF in case an announcement is completed. In this case, the SCSM sends the **PlayAnnouncement** (containing a request for returning a **SpecializedResourceReport** operation as an indication of completion of the operation) or **PromptAndCollectUserInformation** operation with permission of SRF-initiated disconnect to the SRF. In the case of Assisting SSF, the SRF-initiated disconnect cannot be used. In this case, the SCSM transits back to the same state;

- (e4.3'') Continue_SCF_Processing: This is an internal event that takes place when the SCSM finishes the User Interaction and requests the disconnection of bearer connection between the Initiating SSF and SRF by means of SRF initiated disconnect, while no SpecializedResourceReport operation is requested to be returned to the SCF in case the announcement is completed. In this case, the SCSM sends the **PlayAnnouncement** operation (**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. In the case of Assisting SSF, the SRF-initiated disconnect cannot be used. In this case, the SCSM transits to the state 2, **Preparing SSF Instructions**. This event maps into the event (e11);

- (E4.5) Failure_from_SRF: This is an external event caused by the reception of return error for PlayAnnouncement or return error for PromptAndCollectUserInformation operation. In this case, the SCSM transits back to the same state;

- (e4.6) Refresh_Timer_Expired: This is an internal event that takes place on the expiration of $T_{SCF-SSF}$. In this case, the SCSM transmits the **ResetTimer** operation to the Initiating/Assisting SSF, and transits back to the same state; and

- (e4.7) Cancellation_Required: This is an internal event that takes place when the SCSM cancels the previous **PlayAnnouncement** or **PromptAndCollectUserInformation** operation. In this case, the SCSM sends the **Cancel** operation to the Assisting SSF (SSF relay case) or the SRF (Direct SCF-SRF case), and transits back to the same state.

The bearer connection between the SSF and the SRF is disconnected when the SCSM exits from this state.

7.3 SRF application entity procedures

7.3.1 General

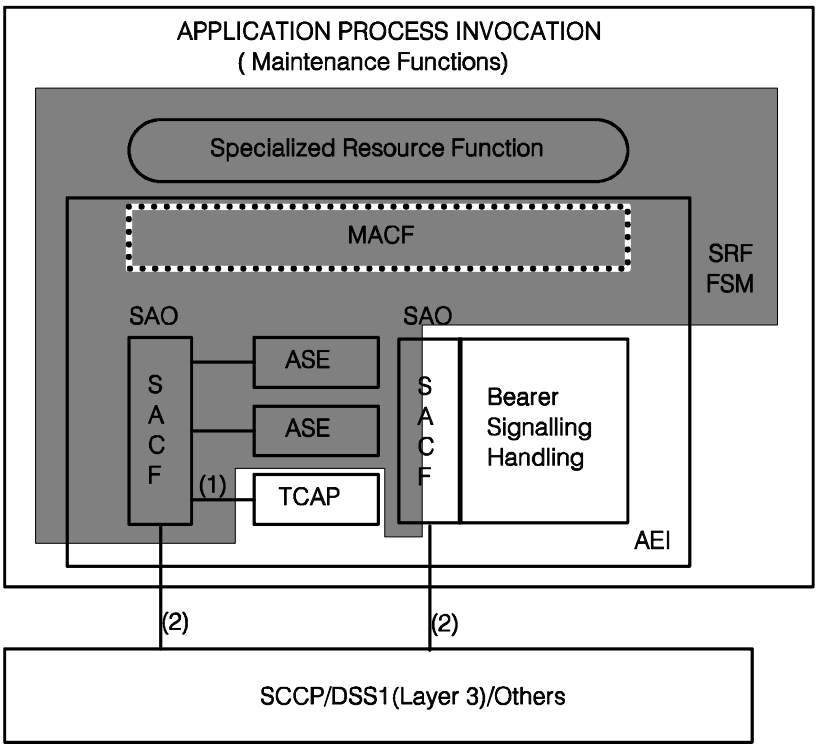
This subclause provides the definition of the SRF AE procedures related to the SRF-SCF interface. The procedures are based on the use of Signalling System No.7; other signalling systems may also be used. Other capabilities may be supported in an implementation-dependent manner in the IP, SSP or SN.

The AE, following the architecture defined in ITU-T Recommendations Q.700 [9] and Q.1400 [13], and ETS 300 287 [5] (ITU-T Recommendation Q.771), includes TCAP and one or more ASEs called TC-users. The following subclauses define the TC-user ASE and SACF/MACF rules, which interface with TCAP using the primitives specified in ETS 300 287 [5] (ITU-T Recommendation Q.771).

The procedure may equally be used with other message based signalling systems supporting the Application Layer structures defined. In case interpretations for the AE procedures defined in the following differ from detailed procedures and the rules for using of TCAP service, the statements and rules contained in the detailed Clauses 9 and 10 shall be followed.

7.3.2 Model and interfaces

The functional model of the AE-SRF is shown in figure 22; the ASEs interface to TCAP (to communicate with the SCF) as well as interface to the maintenance functions. The scope of this ETS is limited to the shaded area in figure 22.



(1) TC-Primitives
(2) N-Primitives

NOTE: The SRF FSM includes several Finite State Machines.

Figure 22: Functional model of SRF AE

The interfaces shown in figure 22 use the TC-user ASE primitives specified in ETS 300 287 [5] (ITU-T Recommendation Q.771) (interface (1)) and N-primitives specified in ETS 300 009 [2] (ITU-T Recommendation Q.711) (interface (2)). The operations and parameters of INAP are defined in Clause 6 of this ETS.

7.3.3 Relationship between the SRF FSM and maintenance functions/bearer connection handling

The primitive interface between the SRF FSM and the maintenance functions is an internal interface and is not subject for Standardization in CS1.

The relationship between the Bearer Connection Handling and the SRF FSM may be described as follows for the case of a call initiated by the SSF: When a call attempt is initiated by the SSF, an instance of an SRF FSM is created.

The SRF FSM handles the interaction with the SCF FSM and the SSF FSM.

The management functions related to the execution of operation received from the SCF are executed by the SRF Management Entity (SRME). The SRME interface the different SRF Call State Models (SRSMs) and the FEAM. Figure 23 shows the SRF FSM structure.

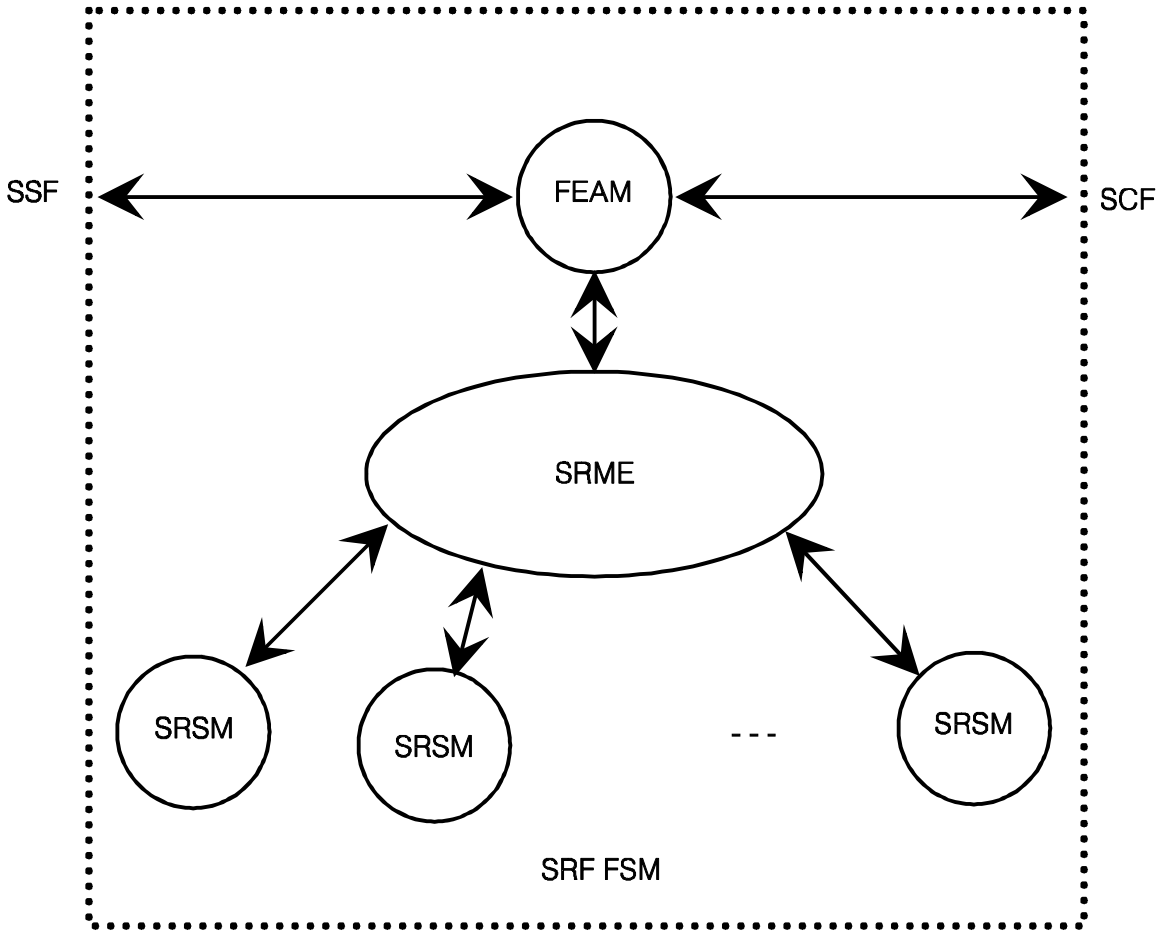


Figure 23: SRF FSM structure

The model associates a FSM with each initial interaction request from the SCF. Thus, multiple initial requests may be executed concurrently and asynchronously by the SRF, which explains the need for a single entity that performs the tasks of creation, invocation, and maintenance of the SRSM objects. In addition to the above tasks, the SRME maintains the dialogues with the SCF and SSF on behalf of all instances of the SRSM.

In particular, the SRME:

1) interprets the input messages from other FEs and translates them into corresponding SRSM events;

2) translates the SRSM outputs into corresponding messages to other FEs; and

3) handles the activity test functionality for the SCF-SRF relationship.

Finally, the FEAM relieves the SRME of low-level interface functions. The FEAM functions include:

1) establishing and maintaining the interfaces to the SSF and SCF;

2) passing (and queueing when necessary) the messages received from the SSF and SCF to the SRME; and

3) formatting, queueing (when necessary), and sending messages received from the SRME to the SSF and SCF.

7.3.4 The SRF Call State Model (SRSM)

The SRSM is presented in figure 24. In what follows, each state is described in a separate subclause together with the events that cause a transition out of this state. Finally, the outputs are presented within smaller rectangles than the states are; unlike the states and events, the outputs are not enumerated.

Each state is discussed in the following subclauses. General rules applicable to more than one state are addressed here.

One component or a sequence of components received in one or more TCAP messages may include a single operation or multiple operations, and it is processed as follows:

- the SRSM processes the operations in the order in which they are received;

- the SRSM examines subsequent operations in the sequence. When a **Cancel** (for **PlayAnnouncement** or **PromptAndCollectUserInformation**) operation is encountered in the sequence in state **User Interaction**, it executes it immediately. In all other cases, the SRSM queues the operations and awaits an event (such an event would be the completion of the operation being executed, or reception of an external event);

- if there is an error in processing one of the operations in the sequence, the SRF FSM processes the error (see below) and discards all remaining operations in the sequence;

- if an operation is not understood or is out of context (i.e. it violates the SACF rules defined by the SRSM) as described above, the SRF FSM processes the error according to the rules given in subclause 10.2 (using TC-U-REJECT or the operation error UnexpectedComponentSequence).

In any state, if there is an error in a received operation, the maintenance functions are informed. Generally, the SRSM remains in the same state in which it received the erroneous operations, however different error treatment is possible in specific cases as described in Clause 8; depending on the class of the operation, the error could be reported by the SRF to the SCF using the appropriate component (see ETS 300 287 [5] (ITU-T Recommendation Q.774)).

In any state, if the dialogue with the SCF (direct SCF-SRF case) is terminated, then the SRSM returns to **Idle** state after ensuring that all resources allocated to the dialogue have been de-allocated.

The SRF shall remain connected to the SSF as long as it has PA operations active or buffered. The resources allocated to the call will be de-allocated when all announcements are completed or when the SSF disconnects the bearer connection (i.e. call party release).

In any state (except **Idle**), if the SSF disconnects the bearer connection to the SRF before the SRF completes the user interaction, then the SRSM clears the call and ensures that all SRF resources allocated to the call have been de-allocated. Then it transits to the **Idle** state.

The SRSM has an application timer, T_{SRF}, whose purpose is to prevent excessive call suspension time. This timer is set when the SRF sends Setup Response bearer message to the SSF (SSF relay case) or the **AssistRequestInstructions** operation (Direct SCF-SRF case). This timer is stopped when a request is received from the SCF. The SRF may reset T_{SRF} on transmission of the **SpecializedResourceReport** operation or the return result for the **PromptAndCollectUserInformation** operation when there is no queued **User Interaction** operation. On the expiration of T_{SRF}, the SRSM transits to the **Idle** state ensuring that all SRF resources allocated to the call have been de-allocated.

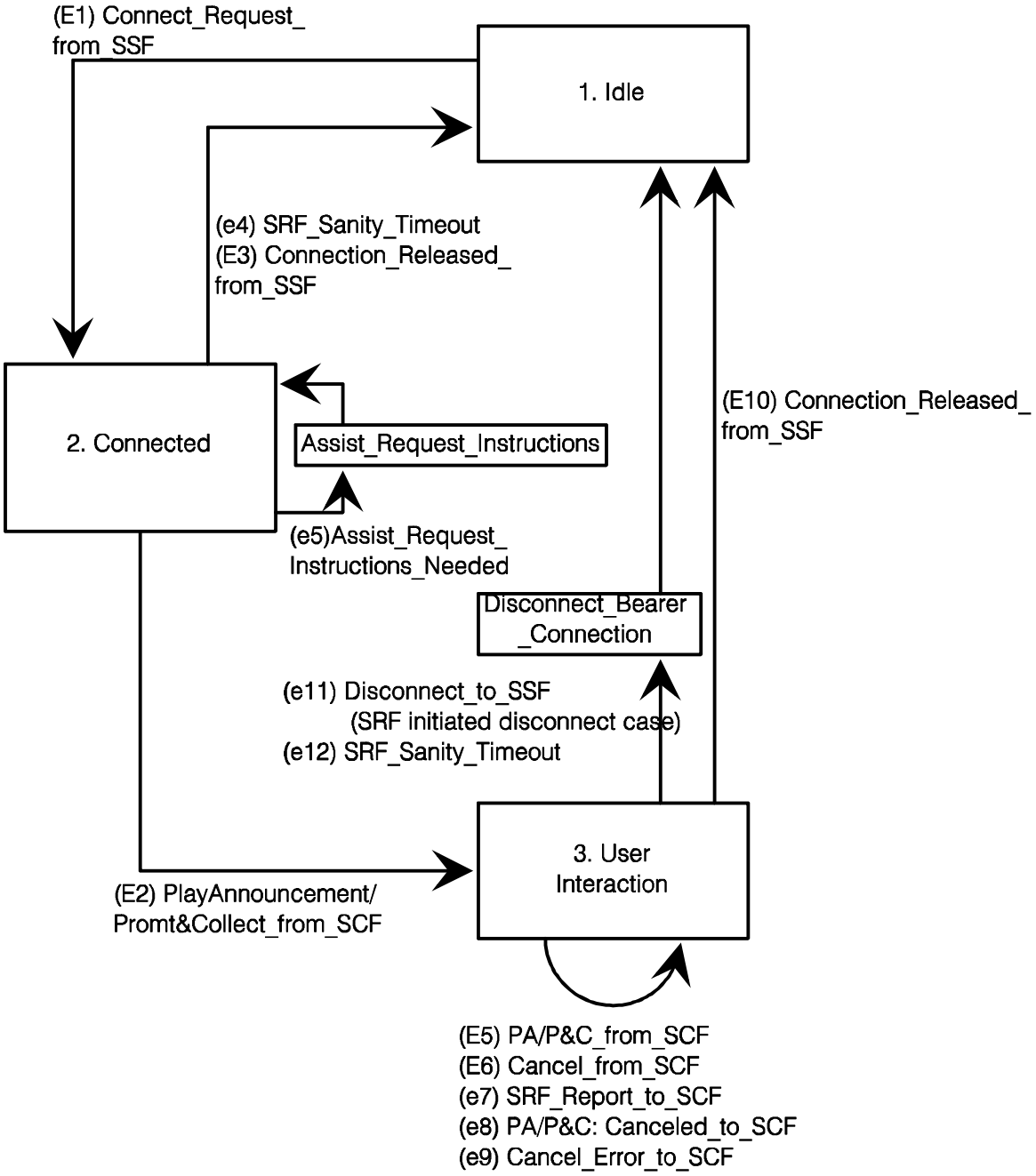


Figure 24: SRSM FSM

7.3.4.1 State 1: "Idle"

The **Idle** state represents the condition prior to, or at the completion of, an instance of user interaction. This state is entered as a result of events E3, e4, E10, e11 and e12. It is exited as a result of event E1.

- (E1) Connect_request_from_SSF: this event corresponds to a bearer signalling connection request message from the SSF. The details of the bearer signalling state machine related to establishing the connection are not of interest to the FSM. The SRSM goes to state "**Connected**";

- (E3) Connection_Released_from_SSF: this event takes place when the SRSM receives a Release message from the SSF in **Connected** state. The SRSM goes to state "**Idle**";

- (e4) SRF_sanity_timeout: this event occurs when the SRSM has been in **Connected** state for a network-operator-defined period of time (timer T_{SRF}) without having a **PlayAnnouncement/**

PromptAndCollectuserinformation (PA/P&C) operation to execute. The SRF initiates a bearer channel disconnect sequence to the SSF using the applicable bearer channel signalling system. The SRSM goes to state "**Idle**";

- (E10) Connection_Released_from_SSF: this event takes place when the SRSM receives a Release message from the SSF in **User Interaction** state. The SRSM goes to state "**Idle**";

- (e11) Disconnect_to_SSF: this event occurs when the SCF has enabled SRF initiated disconnect by the last PA/P&C from SCF (E2) or (E5) with the parameter. The SRSM initiates a bearer channel disconnect sequence to the SSF using the applicable bearer channel signalling system after sending last **SpecializedResourceReport** operation to the SCF (e7). The SRSM goes to state "**Idle**"; and

- (e12) SRF_Sanity_Timeout: this event occurs when the SRSM has been in **User Interaction** state for a network-operator-defined period of time (timer T_{SRF}) without having a **PA/P&C** operation to execute. The SRF initiates a bearer channel disconnect sequence to the SSF using the applicable bearer channel signalling system. The SRSM goes to state "**Idle**".

7.3.4.2 State 2: "Connected"

This state represents the condition of the SRSM when a bearer channel has been established between a user and the SRF but the initial **PA/P&C** has not yet been received (e.g. when **Establish Temporary Connection** procedures are used). The method used to provide this bearer channel is not of interest in the FSM.

- (E1) Connect_request_from_SSF: this event corresponds to a bearer signalling connection request message from the SSF in the **Idle** state. The details of the bearer signalling state machine related to establishing the connection are not of interest in the SRF FSM. The SRSM goes to state "**Connected**";

- (E2) PA/P&C_from_SCF: this event takes place when the first **PlayAnnouncement** or **PromptAndCollectUserInformation** operation(s) from the SCF is received. The SRSM goes to state "**User Interaction**";

- (E3) Connection_Released_from_SSF: this event takes place when the SRF receives a release message from the SSF. The SRSM goes to state "**Idle**";

- (e4) SRF_sanity_Timeout: this event occurs when the SRSM has been connected for a network operator defined period of time (timer T_{SRF}) without having a **PA/P&C** operation to execute. The SRSM initiates a bearer channel disconnect sequence to the SSF using the applicable bearer channel signalling system. The SRSM goes to state "**Idle**";

- (e5) Assist_Request_Instructions_Needed: this event occurs when the **AssistRequestInstructions** operation is sent from the SRSM to the SCF in the absence of a **PA/P&C** event (E2) initiated by the presence of a **PA/P&C** operation concatenated with the setup request from SSF (E1) (Direct SCF-SRF case). No state change occurs as a result of this event.

7.3.4.3 State 3: "User Interaction"

The **User Interaction** state indicates that communication is occurring between the user and the SRF via the bearer channel established at the **Connected** state. This state is entered as a result of event E2. It is exited as a result of events E10, e11 and e12. Events E5, E6, e7, e8 and e9 do not cause a state change. Event E5 also represents additional PA/P&C operations which are buffered as discussed in the procedures.

- (E2) and (E5) PA/P&C_from_SCF: this event takes place when an initial or subsequent **PlayAnnouncement** or **PromptAndCollectUserInformation** operation(s) from the SCF is received. The SRSM goes to state "**User Interaction**" on the first (E2). The SRSM remains in state "**User Interaction**" for subsequent (E5)s;

- (E6) Cancel_from_SCF (for **PA/P&C**): this event takes place when the corresponding **PlayAnnouncement** or **PromptAndCollectUserInformation** operation is received from the SCF. The indicated interaction is terminated if it is presently running, otherwise it is deleted from the buffer. The SRSM remains in state "**User Interaction**";

- (e7) SRF_Report_to_SCF: this event takes place when a **SpecializedResourceReport** operation or a return result for **PromptAndCollectUserInformation** operation is sent to the SCF. The SRSM remains in state "**User Interaction**";

- (e8) PA/P&C_Cancelled_to_SCF: this event takes place when the **PA/P&C** error caused by the **Cancel** (for **PlayAnnouncement** or **PromptAndCollectUserInformation**) operation is sent to the SCF. This event represents the successful cancellation of an active or buffered **PA/P&C** operation. The SRSM remains in state "**User Interaction**";

- (e9) Cancel_Error_to_SCF: this event takes place when the **Cancel** error (for **PlayAnnouncement** or **PromptAndCollectUserInformation**) is sent to the SCF. This event represents the unsuccessful cancellation of a **PA/P&C** operation. The SRSM remains in state "**User Interaction**";

- (E10) Connection_Released_from_SSF: this event takes place when the SRSM receives a release message from the SSF. The SRSM goes to state "**Idle**";

- (e11) Disconnect_to_SSF: this event occurs when the SCF has enabled SRF initiated disconnect with the last **PA/P&C** from SCF (E2) or (E5). The SRSM initiates a bearer channel disconnect sequence to the SSF using the applicable bearer channel signalling system after sending last **SpecializedResourceReport** operation or return result on **PromptAndCollectUserInformation** operation to the SCF. The SRSM goes to state "**Idle**";

- (e12) SRF_sanity_timeout: this event occurs when the SRSM has been connected for a network operator defined period of time (timer T_{SRF}) without having a **PA/P&C** operation to execute. The SRSM initiates a bearer channel disconnect sequence to the SSF using the applicable bearer channel signalling system. The SRSM goes to state "**Idle**".

In addition to these explicitly marked transitions, failure of a user-SRF bearer connection will cause the SRSM to transit to **Idle** from any state. These transitions are not shown in figure 24 for the purpose of visual clarity.

7.3.5 Example SRF control procedures

This subclause provides a detailed description of the SRF procedures. Arrow diagrams are used for the description of the connect, interaction with the end user, and disconnect stages.

The SRF control procedures are based on various physical allocation patterns of SRF. The various control procedures are described in this subclause in accordance with the example physical scenarios of protocol architecture in subclause 4.1.

The Service Assist and Hand-off procedures based on the physical scenarios are also described in this subclause as examples.

NOTE: Through this subclause, bearer connection control signalling messages are used for explanatory purposes, and are not subject for standardization. The terms used for bearer connection control signalling messages only represent the functional meaning.

7.3.5.1 SRF connect procedures

7.3.5.1.1 SRF connect physical procedures

Several procedures are required for different physical scenarios.

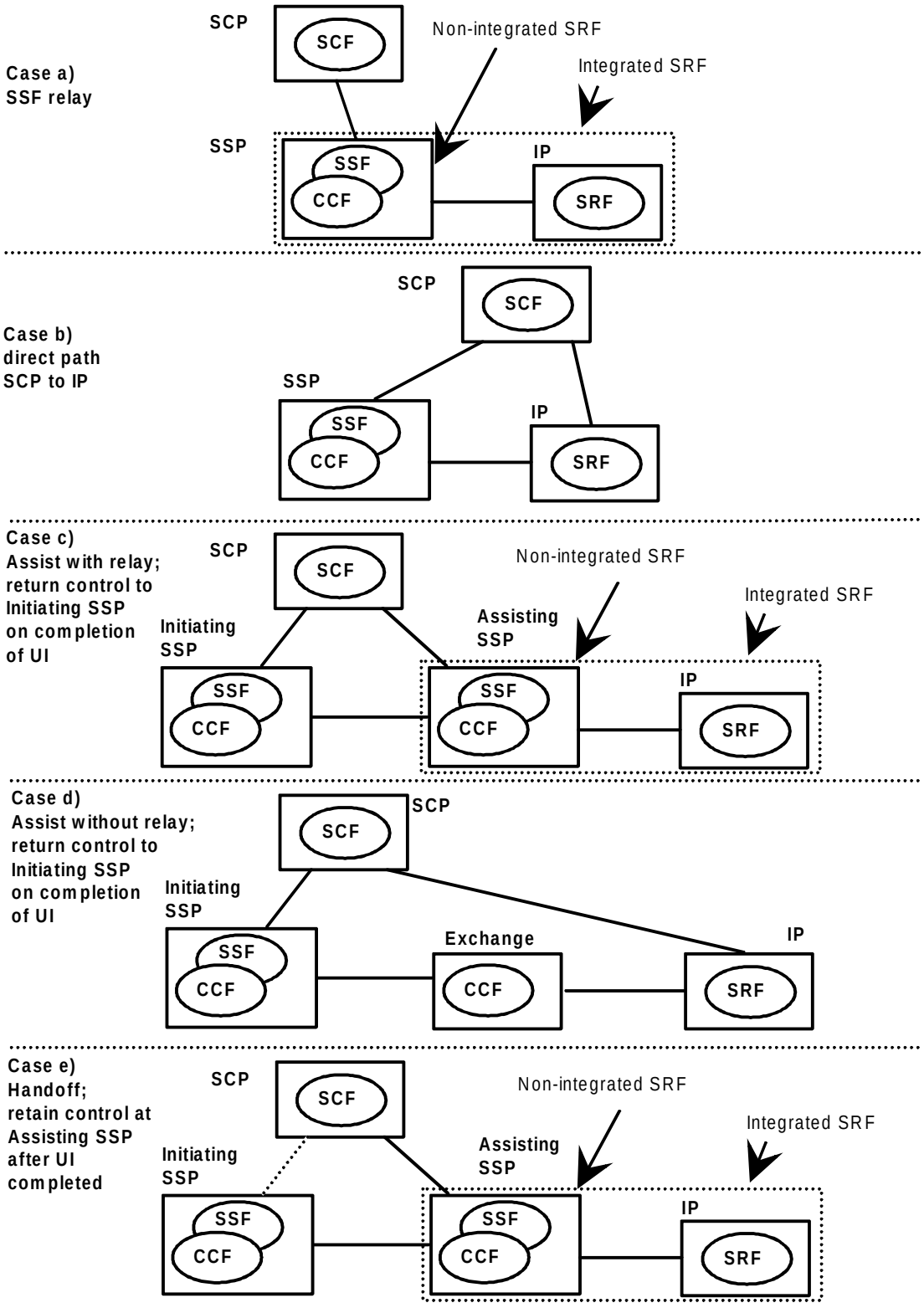


Figure 25: Physical scenarios

The cases to be covered are described below and illustrated in figure 25:

a) the IP is integrated into the SSP, or directly attached to the SSP that is interacting with the SCP but the operations of the SCP to the IP are relayed via the SSP which performs any needed protocol conversion;

b) the IP is directly attached to the SSP that is interacting with the SCP but the operations of the SCP to the IP are sent directly to the IP without SSP relaying involved;

c) the IP is integrated into another SSP, or directly attached to another SSP, than the one that is interacting with the SCP but the operations of the SCP to the IP are relayed via the second SSP (called the "Assist" method), and on completion of the user interaction, control is returned to the first SSP;

d) the IP is directly attached to another node than the SSP that is interacting with the SCP but the operations of the SCP to the IP are sent directly to the IP without SSP relaying involved (called the "Assist" method, but with a variation on the physical connectivity of the entities involved), and on completion of the user interaction, control is returned to the first SSP; and

e) the IP is attached to another SSP and on completion of the user interaction, control of the call is retained at that SSP (called the "Hand-off" approach).

In each of the above cases, the operations between the SCP and the SSP may be Signalling System No.7 TCAP-based; the messaging between the SSP and the IP when the SSP does relaying may be DSS1 using the Facility IE (in this case, the SSP would have to do protocol conversion from Signalling System No.7 TCAP to DSS1 Facility IE for the operations and responses it relayed between the SCP and the IP); the direct messaging between the SCP and the IP may be Signalling System No.7 TCAP based; and bearer control signalling may be any system.

Each of the scenarios will now be examined using arrow diagrams.

Case a) is illustrated in figure 26. For the integrated IP/SSP, the internal activities of the node can still be modelled in this way, but the details of how this is achieved are left to the implementor. This approach makes it unnecessary for the SCP to distinguish between integrated and external but directly connected IPs. See also a note on the possibility of concatenating the first user interaction operation with the **ConnectToResource** operation discussed in the subclause on user interaction below. The establishment of the SCF-SRF relationship in this case is implicit.

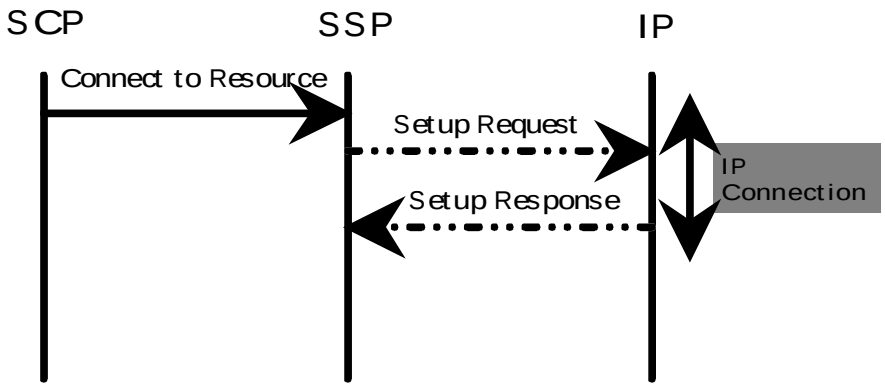


Figure 26: Connection to integrated or external IP with SSP relay of IP operations

Case b) requires that the IP indicate to the SCP that it is ready to receive operations. The establishment of the SCF-SRF relationship is explicit. It is necessary to convey a correlation ID to ensure that the transaction established between the SCP and the IP can be correlated to the bearer connection setup as a result of the preceding operation of the SCP to the SSP.

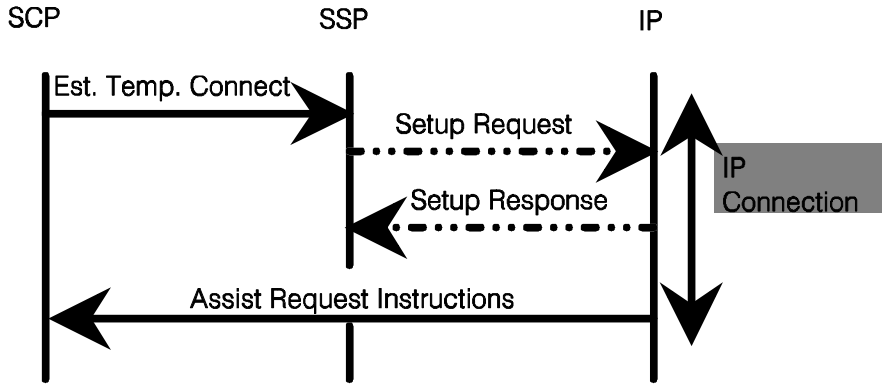


Figure 27: Connection to IP with direct link to SCP, IP initiates interaction with SCP

Case c) requires that a transaction be opened with the Assisting SSP so that it may relay operations from the SCP to the IP (integrated or external). Once the bearer control signalling has reached the assisting SSP, it triggers on the identity of the called facility, and initiates an interaction with the SCP that has requested the assistance (it would also be possible to trigger on other IEs such as the incoming address). The bearer control signalling shall contain information to identify the SCP requesting the assistance, and a correlation ID. This information may be hidden in the address information in such a way that non-message based signalling systems may also be used to establish the bearer connection to the assisting SSP. After the **AssistRequestInstructions** is received by the SCP, the procedures are the same as case a). Figure 28 illustrates the preamble involved.

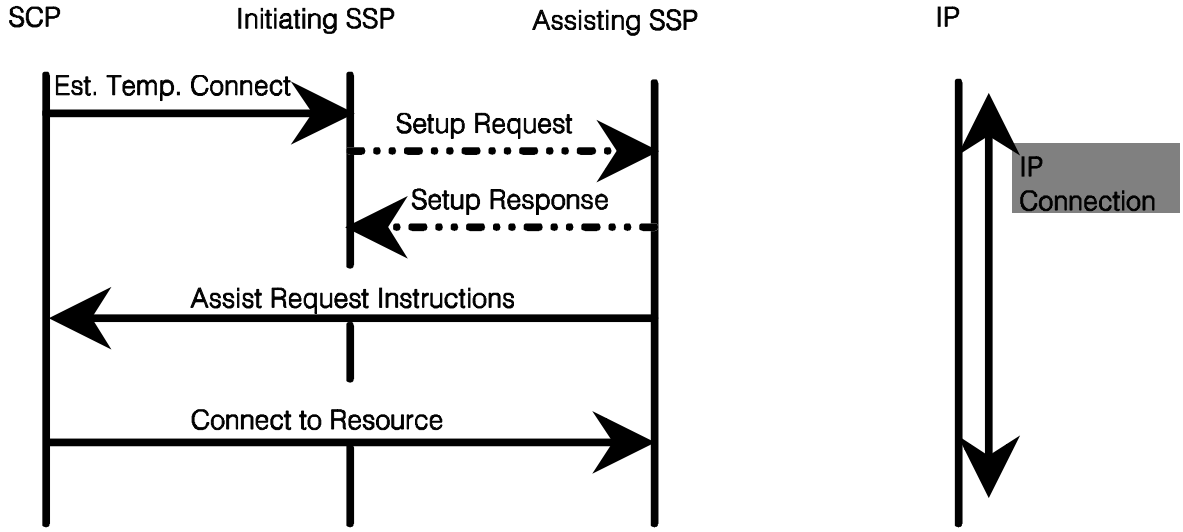


Figure 28: Preamble for Assist case with integrated IP or external IP and SSP relay of SCP-IP messages

Case d) does not require the establishment of a second transaction from the assisting exchange, hence it need not be an SSP. This then becomes a preamble to the procedure shown in figure 27 as shown in figure 29.

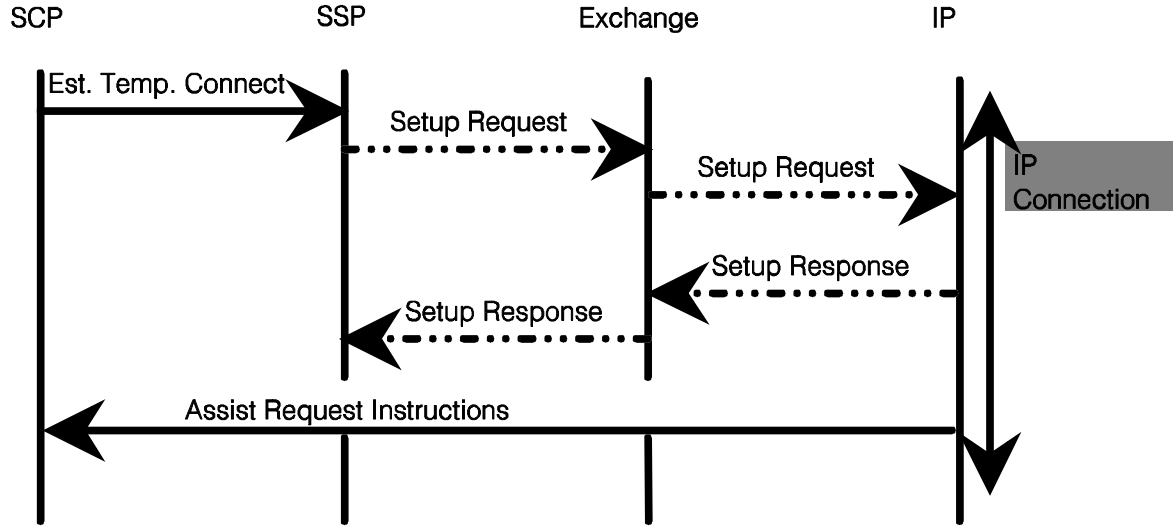


Figure 29: Preamble for Assist case with external IP and direct SCP-IP messaging

Case e) merely requires the sending of an operation to the first SSP to route the call to the handed-off SSP, and then figure 26 applies at handed-off SSP. This is shown in figure 30. The activity at handed-off SSP represents a new interaction with the SCP and **AssistRequestInstructions** is used. Once the bearer control signalling has reached the assisting SSP, it triggers on the identity of the called facility, and initiates an interaction with the SCP that has requested the assistance (it would also be possible to trigger on other IEs such as the incoming address). The bearer control signalling shall contain information to identify the SCP requesting the assistance, and a correlation ID. This information may be hidden in the address information in such a way that non-message based signalling systems may also be used to establish the bearer connection to the assisting SSP.

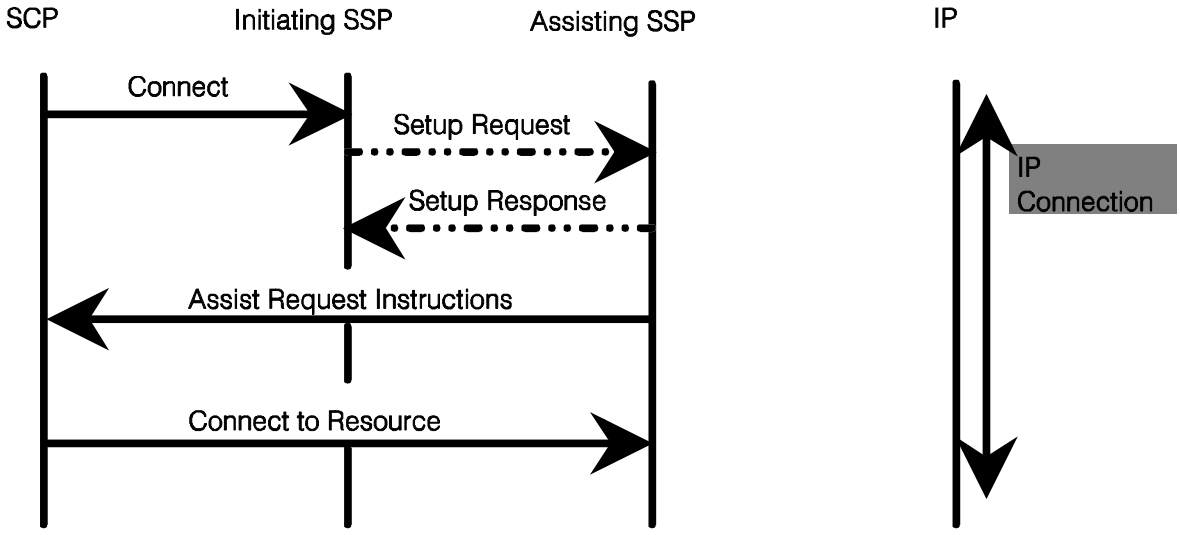


Figure 30: Preamble for hand-off case

7.3.5.2 SRF end user interaction procedures

The end user interaction procedures allow:

- the sending of one or multiple messages to the end user by using the **PlayAnnouncement** operations;
- a dialogue with the end user by using one or a sequence of **PromptAndCollectUserInformation** operations;
- a combination of the above; and
- cancellation of a **PlayAnnouncement** or **PromptAndCollectUserInformation** operations by using a generic **Cancel** operation.

7.3.5.2.1 Play Announcement/Prompt and Collect user information (PA/P&C)

There are only two physical scenarios for user interaction:

a) the SSP relays the operations from the SCP to the IP and the responses from the IP to the SCP (SSF relay case); and

b) the operations from the SCP to the IP and the responses from the IP are sent directly between the SCP and the IP without involving the SSP (Direct SCF-SRF case).

Case a) is illustrated in figure 31 below.

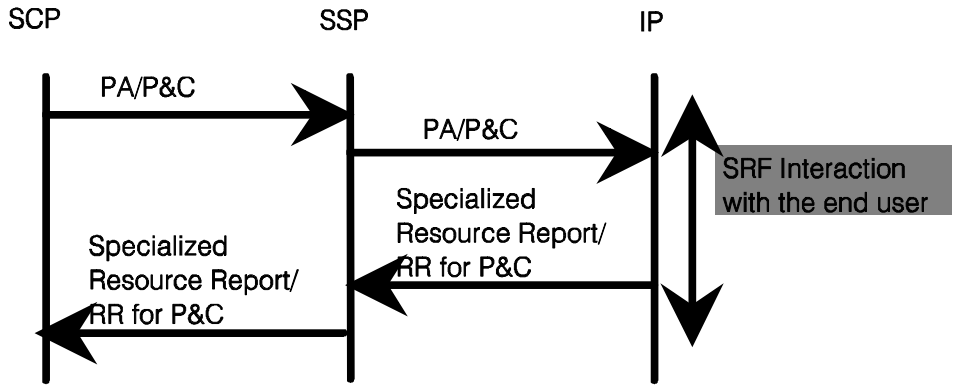


Figure 31: SSP Relay of user interaction operations and responses

Case b) is illustrated in figure 32 below.

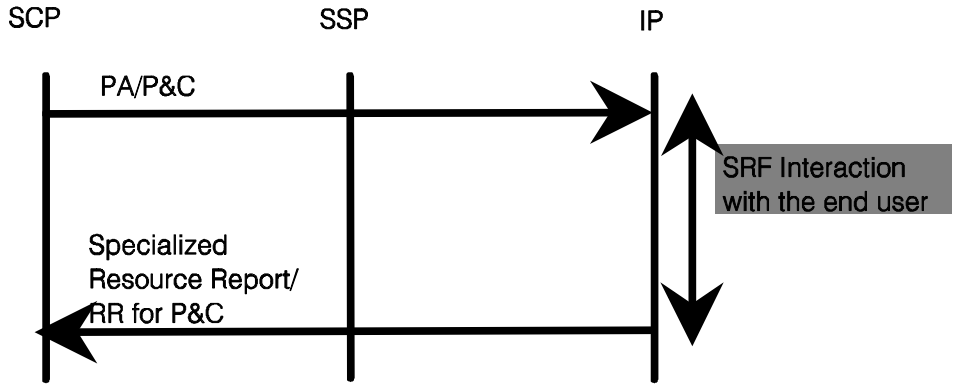


Figure 32: Direct SCF-SRF of user interaction operations and responses

It is also necessary to consider the capability of Signalling System No.7 TCAP to concatenate several Invoke PDUs in one message. This capability allows, for the scenario in figure 26, the **ConnectToResource** and the first **PA/P&C** to be carried in one message. This has some advantages in this physical scenario, such as reduced numbers of messages, and possibly better end-user perceived performance.

7.3.5.3 SRF disconnection procedures

The disconnection procedures are controlled by the SCF and the procedure used is selected based on the needs of the service being executed. The bearer disconnection procedure selected by the SCF is to either allow the SRF to disconnect on completion of user interaction, or to have the SCF explicitly order the SSF to disconnect.

SRF disconnect does not cause disconnection by the SSF/CCF back to the end user terminal unless the transaction with the SCF has been terminated, indicating the user interaction completed the call. The SSF/CCF recognizes that a connection to an SRF is involved because the operations from the SCF for this purpose are distinct from the operations that would be used to route the call towards a destination. There is no impact on bearer signalling state machines as a result of this since incoming and outgoing bearer signalling events are not simply transferred to each other, but rather are absorbed in call processing, and regenerated as needed by call processing. Therefore, to achieve the desired functionality, call processing need simply choose not to regenerate the disconnect in the backward direction. Figure 33 illustrates this concept.

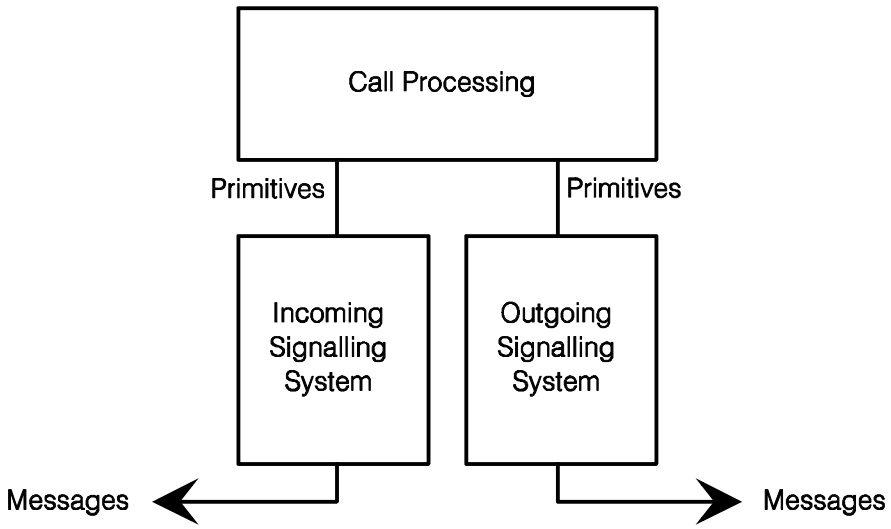


Figure 33: Relationship of incoming and outgoing signalling systems to call processing

As for the SRF connection procedures, the SRF disconnection is affected by the physical network configuration.

In order to simplify the interface between the SCF and the SRF, a number of assumptions are made. The assumptions, and the resulting rules, result in unambiguous procedures from both the SCF and the SRF points of view. The rules, presented below, refer to the SRF originated disconnect, or "SRF Initiated Disconnect", and to the SCF originated disconnect, or "SCF Initiated Disconnect". While other scenarios are possible, they are not included because they either duplicate the functionality presented below or they otherwise do not add value from a service perspective:

a) if a series of **PA/P&C** operations are to be executed by the same SRF, then SRF disconnect is inhibited for all but the last and may be inhibited on the last **PA/P&C**. When a subsequent **PA/P&C** is received, it is buffered until the completion of any preceding **PA/P&C;**

b) a generic **Cancel** operation terminates the indicated **PA/P&C** if it is being executed by the SRF, but does not disconnect the SRF. If the **Cancel** operation is for a buffered **PA/P&C**, that **PA/P&C** is discarded, but the current and any buffered **PA/P&C**s are executed. An SRF interacts with one user only and therefore cancelling a **PA/P&C** only affects the user to which the SRF is connected;

c) the SCF shall either explicitly order "Disconnect" or enable SRF initiated disconnect at the end of the **PA/P&C**. An SRF left connected without a **PA/P&C** to execute may autonomously disconnect if it has not received any **PA/P&C** operations within a defined time limit (this could occur, for example, after an **EstablishTemporaryConnection** which is not followed within a reasonable time period by a **PA/P&C** operation). This sanity timing value will depend on the nature of the interaction the SRF supports and should be selected by the network operator accordingly;

d) when SRF initiated disconnect is enabled in a **PA/P&C**, then the SRF shall disconnect on completion of the user interaction;

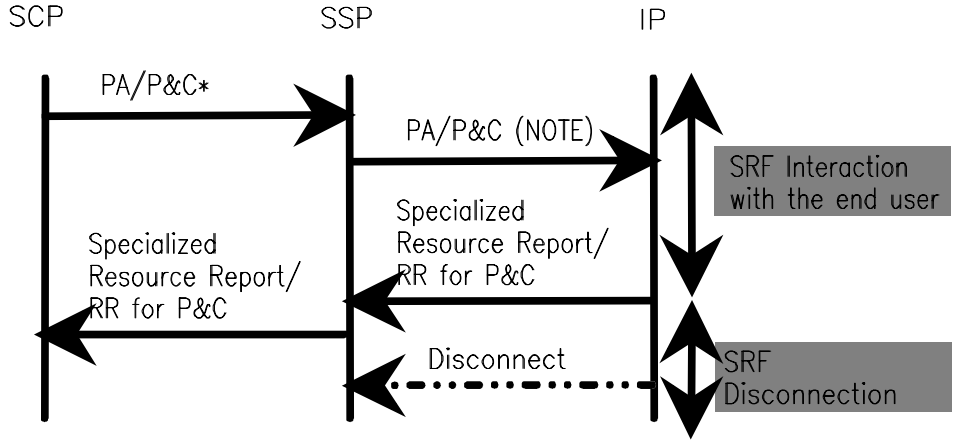
e) when SRF initiated disconnect is not enabled, the SCF shall ask the SRF to inform it of the completion of the User Interaction using the **SpecializedResourceReport** operation for "announcement complete" or using the return result for the **PromptAndCollectUserInformation** operation;

f) if the user disconnects, the SRF is disconnected and the SSF releases resources and handles the transaction between the SSF and the SCF as specified in ITU-T Recommendation Q.1214 [11] and in this ETS. The SRF discards any buffered operations and returns its resources to idle. The relationship with the SCF is terminated;

g) when the SCF explicitly orders the SSF to disconnect by **DisconnectForwardConnection** operation, the SSF releases the bearer connection to the SRF, and returns to the "**Waiting for Instructions**" state. No operation reporting SRF disconnect from the SSF to the SCF is required.

7.3.5.3.1 SRF initiated disconnect

The SRF disconnect procedure is illustrated in figure 34. The SRF disconnect is enabled by the SCF within a **PA/P&C** operation. When the SRF receives a **PA/P&C** enabling disconnection, it completes the dialogue as instructed by the **PA/P&C**, and then initiates the SRF initiated disconnection using the applicable bearer control signalling. The SSF/CCF knows that it is an SRF disconnecting and does not continue clearing the call toward the end user. The SSF returns to the "**Waiting for Instructions**" state and executes any buffered operations. In the Hand-off case, the SSP shown in figure 34 is the "handed-off" SSP.



NOTE: Disconnect from SRF is forbidden.

Figure 34: SCF disconnect for Local, Embedded and Hand-off scenarios

For the Assisting SSF case, the SRF initiated disconnect procedures are not used because the Assisting SSF remains in the "**Waiting for Instructions**" state and does not propagate the disconnection of the bearer connection to the Initiating SSF. The SCF initiated disconnect procedures described in the following subclause are used for the Assisting SSF case.

For the Direct SCF-SRF case, the procedures also work in the same manner. The SRF disconnect is enabled by the SCF within a **PA/P&C** operation. When the SRF receives a **PA/P&C** enabling disconnection, it completes the dialogue as instructed by the **PA/P&C**, and then initiates the SRF initiated disconnection using the applicable bearer control signalling. The Initiating SSF/CCF knows that it is an SRF disconnecting and does not continue clearing the call toward the end user. The Initiating SSF returns to the "**Waiting for Instructions**" state and executes any buffered operations.

7.3.5.3.2 SCF initiated disconnect

The SCF initiated disconnect procedure is illustrated in figure 35 (bearer messages are shown with dotted lines). The figure shows only the Assisting SSF case, and the Direct SCF-SRF case is not shown. To initiate the SCF initiated disconnection of the SRF, the SCF shall request and receive a reply to the last **PA/P&C** operation requested. The **SpecializedResourceReport** operation contains an "announcement complete" and return result for **P&C** contains "collected information."

The SCF initiated disconnect uses an operation called **DisconnectForwardConnection**. Once the **DisconnectForwardConnection** operation is received by the SSF, it will initiate a "release of bearer channel connection" between the PEs containing the SSF and SRF, using applicable bearer control signalling. Since the SCF (which initiates the disconnect), the SSF (which instructs bearer signalling to disconnect) and the SRF (which receives disconnect notification via bearer signalling) are aware that disconnect is occurring, they are synchronized. Therefore, a "pre-arranged" end may be used to close the transaction. This does not preclude the use of explicit end messages for this purpose.

For Assisting SSF case, the initiating SSP, on receipt of the Disconnect Forward Connection from the SCP, disconnects forward to the assisting SSP, and this disconnection is propagated to the IP. The initiating SSP, knowing that the forward connection was initiated as the result of an Establish Temporary Connection, does not disconnect back to the user but returns to the "Waiting for Instructions" state.

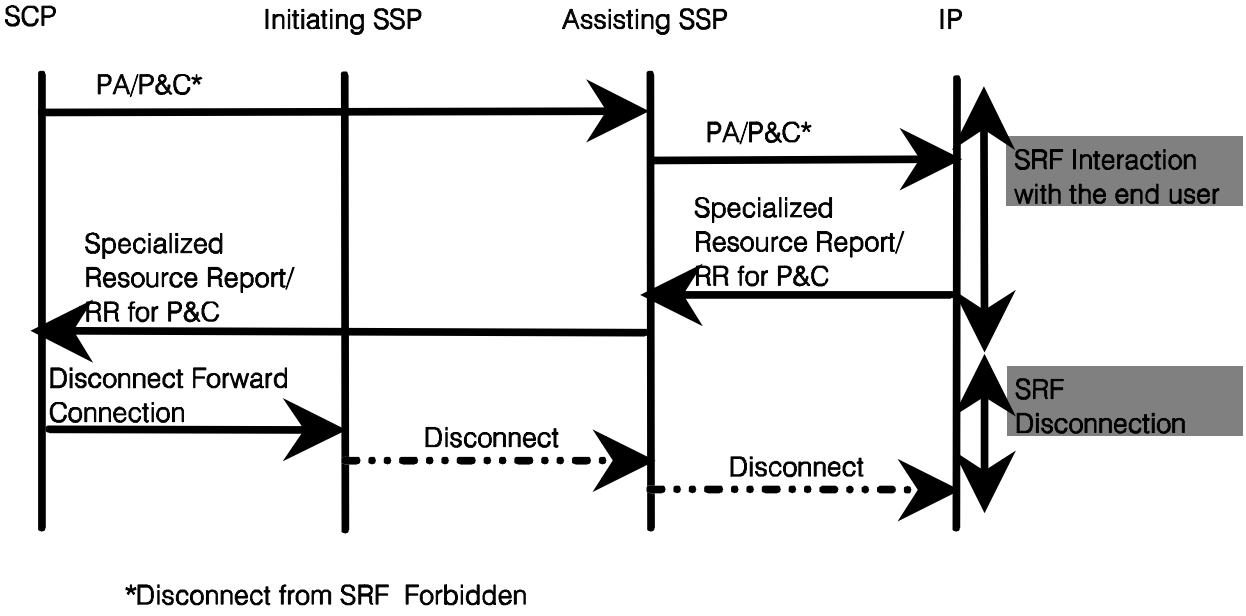


Figure 35: SCF initiated disconnect for Assist scenario

7.3.5.4 Examples illustrating complete user interaction sequences

The following figures and their accompanying tables provide examples of complete sequences of user interaction operations covering the three stages:

- connect the SRF and the end user (bearer connection) and establish the SCF-SRF relationship;
- interact with the end user;
- disconnect the SRF and the end user (bearer connection) and terminate the SCF-SRF relationship.

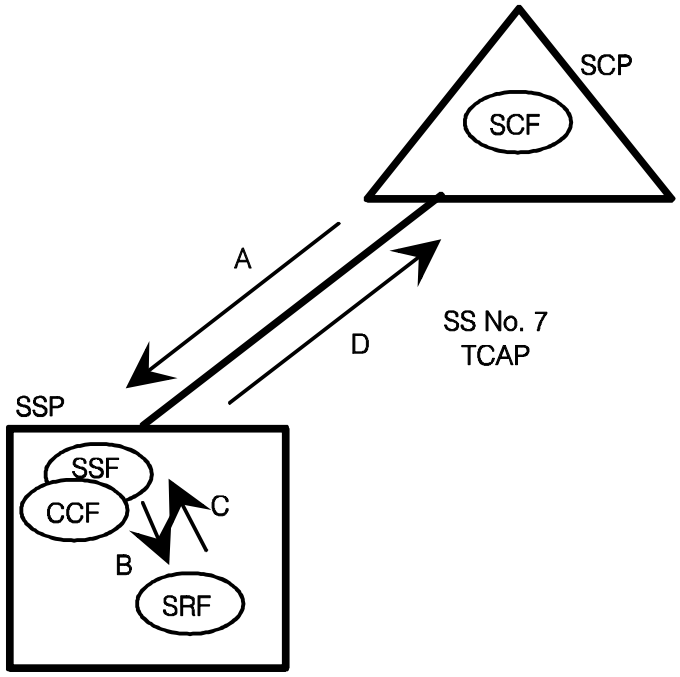


Figure 36: SSP with integrated SRF

In figure 36 above, the SSP with an integrated (or embedded) SRF, the procedural scenarios can be mapped as given in table 4.

Table 4

Procedure Name	Operations	Protocol Flows
Connect to Resource and first PA/P&C	ConnectToResource; PA/P&C	A
	Setup; PA/P&C	B
User Interaction	PA/P&C	A then B
	SpecializedResourceReport/ RR for P&C	C then D
SRF Initiated Disconnect	SpecializedResourceReport/ RR for P&C	C then D
	Disconnect	C (intra-SSP bearer control)
SCF Initiated Disconnect	SpecializedResourceReport/ RR for P&C	C then D
	DisconnectForwardConnection	A
	Disconnect	B (intra-SSP bearer control)

A simple extension to this integrated case is the configuration where the SRF is located in an intelligent peripheral locally attached to the SSP. The SCP-IP operations are relayed via the SSF in the SSP. This is depicted in figure 37.

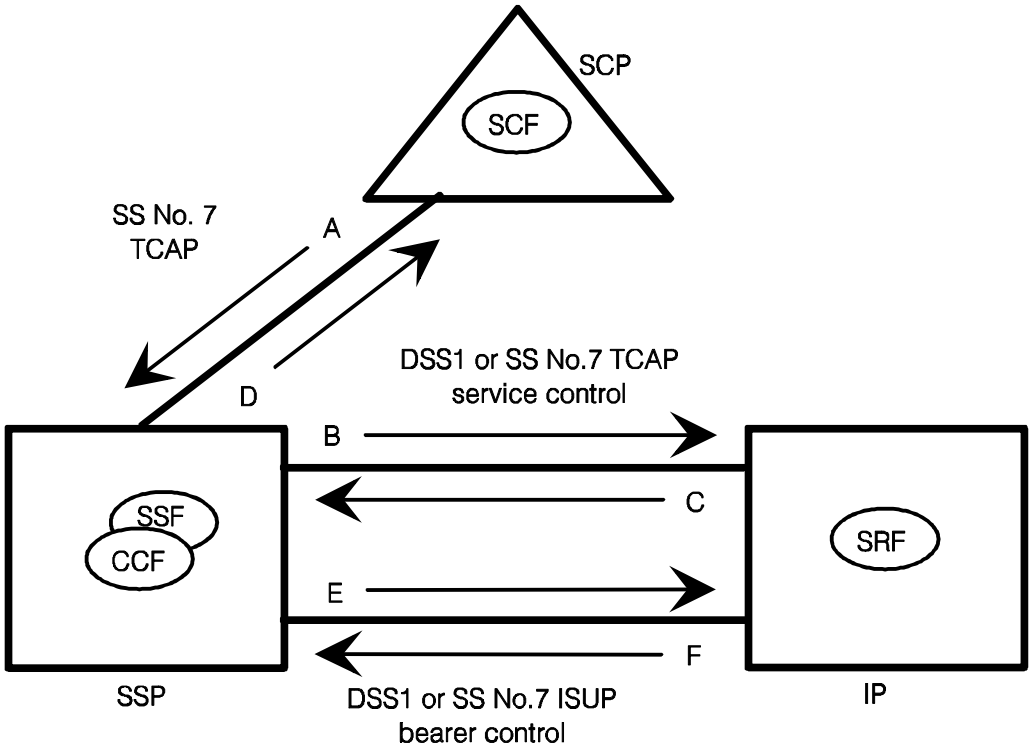


Figure 37: SSP relays messages between SCP and IP

The procedural scenarios for this Relay SSF with an IP (figure 37) can be mapped as shown in table 5.

Table 5

Procedure Name	Operations	Protocol Flows
Connect to Resource and first PA/P&C	ConnectToResource; PA/P&C	A
	If DSS1 used:	
	Setup; PA/P&C	E and B (Facility IE)
	If Signalling System No.7 used:	
	Setup	E
	PA/P&C	B
User Interaction	PA/P&C	A then B
	SpecializedResourceReport/ RR for P&C	C then D
SRF Initiated Disconnect	SpecializedResourceReport/ RR for P&C	C then D
	Disconnect	F
SCF Initiated Disconnect	SpecializedResourceReport/ RR for P&C	C then D
	DisconnectForwardConnection	A
	Disconnect	E

In some cases, the IP may have an Signalling System No.7 or other interface to the controlling SCP. This case is shown in figure 38. The SCP shall correlate two transactions to co-ordinate the activities.

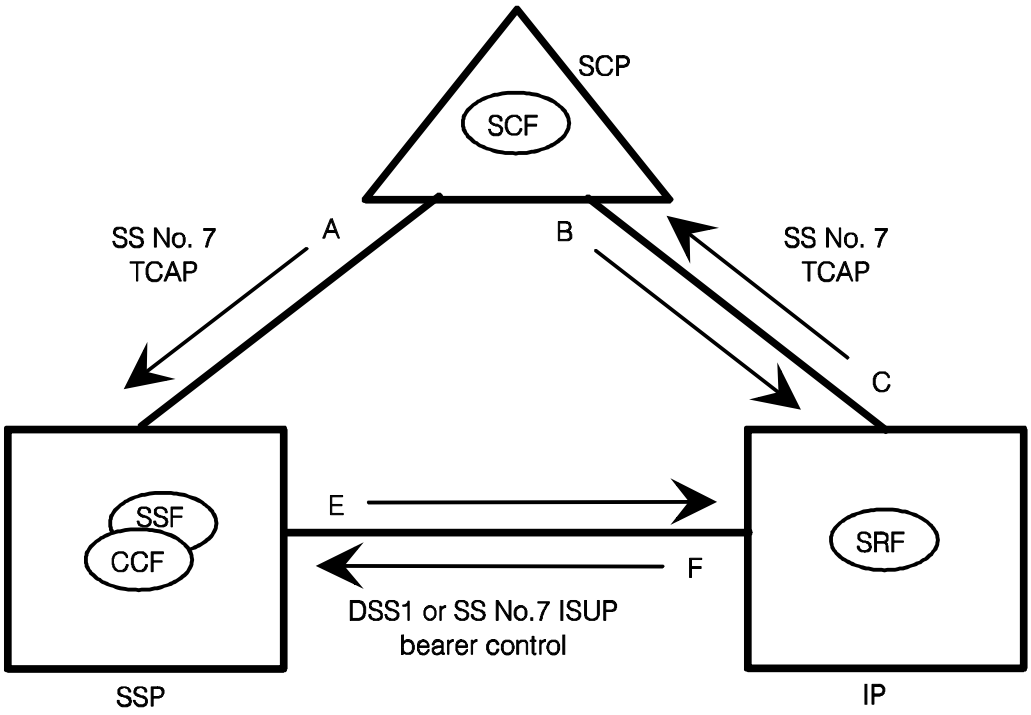


Figure 38: Direct SCP-IP information transfer

In figure 38, the procedural scenarios can be mapped as shown in table 6.

Table 6

Procedure Name	Operations	Protocol Flows
Connect to Resource	EstablishTemporaryConnection	A
	Setup	E
	AssistRequestInstructions	C
User Interaction	PA/P&C	B
	SpecializedResourceReport/ RR for P&C	C
SRF Initiated Disconnect	SpecializedResourceReport/ RR for P&C	C
	Disconnect	F
SCF Initiated Disconnect	SpecializedResourceReport/ RR for P&C	C
	DisconnectForwardConnection	A
	Disconnect	E

The Assisting SSF scenario involves straightforward procedural extensions to the basic cases shown above. One mapping of the assisting SSF case is shown in figure 39. In this case, SRF initiated disconnect cannot be used. Other physical mappings can be derived as described in the text following the figure and its accompanying table.

The integrated SRF and SSF relay case requires a transaction between the SCP and the assisting SSP (figure 39) but the SCP direct case does not since the transaction is directly between the SCP and the IP connected to the remote exchange. In the latter case, any transit exchanges, including the one the IP (SRF) is connected to, are transparent to the procedures.

The SCP shall again correlate two transactions.

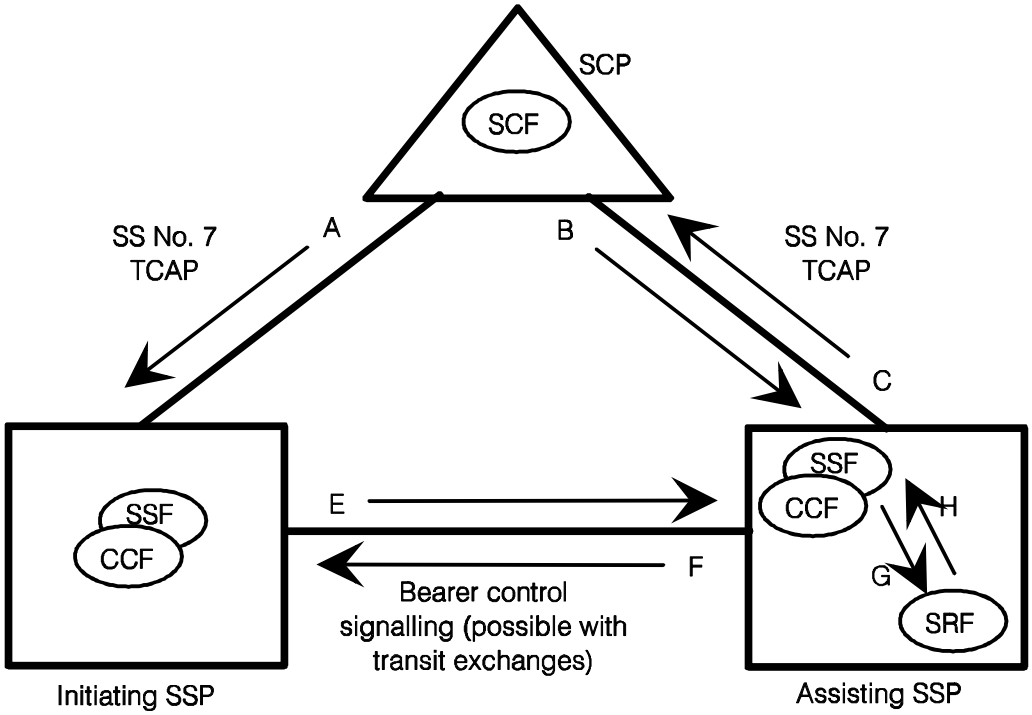


Figure 39: SSP Assist/hand-off (Relay SSP)

In figure 39, the procedural scenarios can be mapped as shown in table 7.

Table 7

Procedure Name	Operations	Protocol Flows
Assist preamble	EstablishTemporaryConnection	A
	Setup	E
	AssistRequestInstructions	C
	ConnectToResource	B
	Setup	G
	ResetTimer	A
User Interaction	PA/P&C	B then G
	SpecializedResourceReport/ RR for P&C	H then C
SCF Initiated Disconnect	SpecializedResourceReport/ RR for P&C	H then C
	DisconnectForwardConnection	A
	Disconnect	E and G (intra-SSP bearer ctrl)

The Assisting SSP case shown in figure 39 can be generalized to cover both the case where the SRF is embedded in Assisting SSP (as shown), and the case where the SRF is locally connected to Assisting SSP. In this latter case, the SRF communication (protocol flows B, C, G, and H) would conform to the physical scenario shown in figure 37.

The service hand-off scenario can similarly be viewed as a sequence consisting of an IN service to route a call from one SSP to another, followed by any one of the previously described physical user interaction scenarios. For describing this scenario, figure 39 can be used also.

7.3.5.4.1 Message sequences for service assist

The following subclause provides additional details on the message sequences for the service assist procedure in figure 39:

1) The SCP, during the processing of a request for instruction, determines that resources remote from the initiating SSP are required and that call processing will continue from the initiating SSP after the remote resources have been used (e.g., the call will be completed to a destination address after information is collected from the calling party). An EstablishTemporaryConnection operation containing the address of the assisting SSP (for routing the call), the ScfID and the CorrelationID (both used for the assisting SSP to establish communication back to the SCP) is sent to the initiating SSP. EstablishTemporaryConnection is used instead of a regular Connect operation because of the nature of the connection to the assisting SSP. The initiating SSP shall be aware that the SCP will ask it to continue in the processing of the call at some point in the future.

   NOTE: The ScfID and CorrelationID may be included in the routing address of the assisting SSP.

   Protocol Flow A

2) The initiating SSP routes the call to the assisting SSP. The ScfID and CorrelationID are sent to the assisting SSP. Existing in-band signalling and Signalling System No.7 information elements (e.g. routing number) could be used to transport this information. The transport mechanism used to send this information between SSPs is independent of the service assist control procedures between the SCF and SSF.

   Protocol Flow E

3) The assisting SSP uses an AssistRequestInstructions operation to establish communication with the SCP. The CorrelationID is sent in the AssistRequestInstructions to allow the SCP to correlate two transactions.

   Protocol Flow C

4) The SCP sends instructions to the assisting SSP based on SL control.

   Protocol Flow B

5) The SCP may need to generate ResetTimer events to the initiating SSP so that it does not time out the call.

   Protocol Flow A

6) When resource functions have been completed, a DisconnectForwardConnection operation is sent to the initiating SSP. This indicates, that the temporary connection to the assisting SSP has to be disconnected.

   Protocol Flow A

7) The initiating SSP sends a message via bearer control signalling to the assisting SSP to close the "assist" transaction.

   Protocol Flow E

8) The call control returns to the initiating SSP.

7.3.5.4.2 Message sequences for hand-off

The following subclause outlines message sequences for the Hand-off procedure using the protocol flows shown in figure 39:

1) The SCP, during the processing of a request for instruction, determines that resources remote from the initiating SSP are required and that call processing need not continue from the initiating SSP after the remote resources have been used (e.g. a terminating announcement will be played). A Connect operation containing the address of the assisting SSP (for routing the call), the ScfID and the CorrelationID (both used for the assisting SSP to establish communication back to the SCP) is sent to the initiating SSP.

   NOTE: The ScfID and CorrelationID may be included in the routing address of the assisting SSP.

   Protocol Flow A

2) The initiating SSP routes the call to the assisting SSP. The ScfID and CorrelationID are sent to the assisting SSP. Existing in-band signalling and Signalling System No.7 information elements (e.g. routing number) could be used to transport this information. The transport mechanism used to send this information between SSPs is independent of the service assist control procedures between the SCF and SSF.

   Protocol Flow E

3) The assisting SSP uses an AssistRequestInstructions operation to establish communication with the SCP. The CorrelationID is sent in the AssistRequestInstructions to allow the SCP to correlate two transactions. AssistRequestInstructions is used instead of a regular request instruction (InitialDP) because the SCP shall associate the AssistRequestInstructions from the assisting SSP/IP with an already active dialogue the SCP has with another SSP.

   Protocol Flow C

4) The SCP sends instructions to the assisting SSP based on SL control.

   Protocol Flow B

5) The call control remains at the assisting SSP.

The same service assist and hand-off procedures can be reused for a direct link to an IP in this and future capability sets.

8 Error procedures

This subclause defines the generic error procedures for the core INAP CS1. The error procedure descriptions have been divided in two subclauses, subclause 8.1 listing the errors related to INAP operations and subclause 8.2 listing the errors related to error conditions in the different FEs which are not directly related to the INAP operations.

8.1 Operation related error procedures

The following subclauses define the generic error handling for the operation related errors. The errors are defined as operation errors in Clause 6. Errors which have a specific procedure for an operation are described in Clause 9 with the detailed procedure of the related operation.

The TCAP services which are used for reporting operation errors are described in Clause 10. All errors which can be detected by the ASN.1 decoder already may have been detected during the decoding of the TCAP message and indicated by the TC error indication "MistypedParameter".

8.1.2 Cancelled

8.1.2.1 General description

8.1.2.1.1 Error description

The error "Cancelled" gives an indication to the SCF that the cancellation, as it was requested by the SCF, of a specific operation, has been successful. The SCF is only able to cancel certain predefined SCF->SRF operations.

8.1.2.2 Operations SCF->SRF

PlayAnnouncement
PromptAndCollectUserInformation

Procedures at invoking entity (SCF)
A) Sending Cancel
precondition: SCSM state 4.1 Waiting for Response from the SRF
postcondition: SCSM state 4.1 Waiting for Response from the SRF
The SCF sends a Cancel after a PlayAnnouncement (PA) or PromptAndCollectUserInformation (P&C) has been sent. The SCF remains in the same state.

B) Receiving Cancelled error
precondition: SCSM state any SL dependent
postcondition: SCSM state any SL dependent
After sending a Cancel operation the SL may continue (e.g. sending more PA or P&C or a DisconnectForwardConnection). The Cancelled error can therefore be received in any state. The treatment is SL dependent.

Procedures at responding entity (SRF)
A) Receiving Cancel
precondition: SRSM state 3 User Interaction
postcondition: SRSM state 3 User Interaction
The indicated PA or P&C is terminated if it is presently executing or deleted from the buffer. If the indicated PA or P&C is already executed this causes a failure ("CancelFailed").

B) Sending Cancel error
precondition: SRSM state 3 User Interaction
postcondition: SRSM state 3 User Interaction
After returning the "Cancelled" error the SRF stays in the same state. The execution of the indicated PA or P&C is aborted, i.e. the SRF remains connected and the next PA or P&C is executed if available.

8.1.3 CancelFailed

8.1.3.1 General description

8.1.3.1.1 Error description

This error is returned by Cancel if the cancelling of an operation, as requested by the SCF, was not successful. Possible failure reasons are:

0 unknownOperation, when the InvokeID of the operation to cancel is not known to SRF (this may also happen in case the operation has already been completed);

1 tooLate, when the invokeID is known but the execution of the operation is in a stadium that it cannot be cancelled anymore. For instance the announcement is finished but the SpecializedResourceReport has not been sent to the SCF yet. The conditions for the occurrence of failure reason "tooLate" may be implementation dependent;

2 operationNotCancellable, when the invokeID points to an operation that the SCF is not allowed to cancel.

8.1.3.1.2 Argument description

```
PARAMETER SEQUENCE {
        problem         [0] ENUMERATED {
                unknownOperation(0),
                tooLate(1),
                operationNotCancellable(2)},
        operation       [1] InvokeID
        }
--  The operation failed to be cancelled.
```

8.1.3.2 Operations SCF->SRF

Cancel

Procedures at invoking entity (SCF)
A) Sending Cancel
precondition: SCSM state 4.1 Waiting for Response from the SRF
postcondition: SCSM state 4.1 Waiting for Response from the SRF
The SCF sends a Cancel after a Play Announcement (PA) or PromptAndCollectUserInformation (P&C) has been sent. It remains in the same state.

B) Receiving CancelFailed error
precondition: SCSM state any SL dependent
postcondition: SCSM state any SL dependent
After sending a Cancel operation the SL may continue (e.g. sending more PA or P&C or a DisconnectForwardConnection). The CancelFailed error can therefore be received in any state. The treatment is SL dependent.

Procedures at responding entity (SRF)
A) Receiving Cancel. However, the indicated PA or P&C is not known, or already executed. This causes a failure, CancelFailed.
precondition: SRSM state 3 User Interaction
postcondition: SRSM state 3 User Interaction
or SRSM state 1 Idle

B) Sending CancelFailed error
precondition: SRSM state 3 User Interaction
or SRSM state 1 Idle
postcondition: SRSM state 3 User Interaction
or SRSM state 1 Idle
After returning the CancelFailed the SRF stays in the same state.

8.1.4 ETCFailed

8.1.4.1 General description

8.1.4.1.1 Error description

ETCFailed is an error from SSF to SCF, indicating the fact that the establishment of a temporary connection to an assisting SSF or SRF was not successful (e.g. receiving a "Backwards Release" after sending the IAM).

8.1.4.2 Operations SCF->SSF

EstablishTemporaryConnection

Procedures at invoking entity (SCF)
A) SCF sends ETC to SSF
precondition: SCSM state 3.1 Determine Mode
postcondition: SCSM state 3.2 Waiting for AssistRequestInstructions

B) SCF receives ETCFailed error from SSF
precondition: SCSM state 3.2 Waiting for AssistRequestInstructions
postcondition: SCSM state 2.1 Preparing SSF Instructions
Error handling depends on the SL, e.g. selecting another SRF or continue the processing of the call.

Procedures at responding entity (SSF)
SSF receives ETC from SCF but the establishment of the connection fails, resulting in the returning of the ETCFailed error to the SCF
precondition: SSF FSM state c Waiting for Instructions
postcondition: SSF FSM state c Waiting for Instructions
No further error treatment.

8.1.5 ImproperCallerResponse

8.1.5.1 General description

8.1.5.1.1 Error description

The format of the user input has been checked by the SRF and does not correspond to the required format as it was defined in the initiating P&C operation.

8.1.5.2 Operations SCF->SRF

PromptAndCollectUserInformation

Procedures at invoking entity (SCF)
A) SCF sends P&C to SRF
precondition: SCSM state 3.1 Determine Mode; P&C will accompany the ConnectToResource
or SCSM state 3.2 Waiting for AssistRequestInstructions; after EstablishTemporaryConnection
or SCSM state 4.1 Waiting for Response from the SRF; if more PAs or P&Cs are active
postcondition: SCSM state 4.1 Waiting for Response from the SRF

B) SCF receives ImproperCallerResponse error from SRF
precondition: SCSM state 4.1 Waiting for Response from the SRF
postcondition: SCSM state 4.1 Waiting for Response from the SRF
Error treatment depends on SL. SCF can initiate new User Interaction or forcing a Disconnect (to SSF).

Procedures at responding entity (SRF)
A) SRF receives P&C
precondition: SRSM state 2 Connected
or SRSM state 3 User Interaction
postcondition: SRSM state 3 User Interaction

B) response from caller is not correct, SRF returns ImproperCallerResponse to SCF
precondition: SRSM state 3 User Interaction
postcondition: SRSM state 3 User Interaction
SRF waits for a new operation from SCF. This may be a new P&C or PA.

8.1.6 MissingCustomerRecord

8.1.6.1 General description

8.1.6.1.1 Error description

The SLP could not be found in the SCF, because the required customer record does not exist, or the requested SLPI, indicated by correlationID in AssistRequestInstructions does not exist anymore.

8.1.6.2 Operations SSF->SCF

AssistRequestInstructions
InitialDP

Procedures at invoking entity (SSF)
A) Sending operation
precondition: SSF FSM state b Trigger processing
or SSF FSM state b' Waiting for Instructions; in case of assist/hand-off
postcondition: SSF FSM state c Waiting for Instructions
or SSF FSM state b' Waiting for Instructions; in case of assist/hand-off

B) SSF receives error "MissingCustomerRecord"
precondition: SSF FSM state c Waiting for Instructions
or SSF FSM state b' Waiting for Instructions; in case of assist/hand-off
postcondition: SSF FSM state a Idle
or SSF FSM state a' Idle; in case of assist/hand-off
The CCF routes the call if necessary (e.g. default routing to a terminating announcement).

Procedures at responding entity (SCF)
The SCSM detects that the required SLP does not exist. The SLPI may not exist anymore (e.g. in case of the operation AssistRequestInstructions), or the SLP may have never existed at all (i.e. the customer record in the SCF does not exist, e.g. in case of TDPs a SLP is attempted to be invoked). The error parameter MissingCustomerRecord is used to inform the invoking entity of this situation. The maintenance functions are informed.

8.1.6.3 Operations SRF->SCF

AssistRequestInstructions

Procedures at invoking entity (SRF)
A) Sending operation
precondition: SRSM state 2 Connected
postcondition: SRSM state 2 Connected

B) SRF receives error "MissingCustomerRecord"
precondition: SRSM state 2 Connected
postcondition: SRSM state 1 Idle
SRF initiated Disconnect.

Procedures at responding entity (SCF)
The SCSM detects that the required SLP does not exist (anymore). The establishment of a connection between the SSF and the SRF took too long or the correlationID was invalid. In both cases the requested SLP cannot be found. The error parameter MissingCustomerRecord is used to inform the invoking entity of this situation. The maintenance functions are informed.

8.1.7 MissingParameter

8.1.7.1 General description

8.1.7.1.1 Error description

There is an error in the received operation argument. The responding entity cannot start to process the requested operation because the argument is incorrect: an expected optional parameter which is essential for the application is not included in the operation argument.

8.1.7.2 Operations SCF->SSF

Non-call Associated
ActivateServiceFiltering

Call Associated/Non-call Processing

ApplyCharging	CallInformationRequest
FurnishChargingInformation	RequestNotificationChargingEvent
RequestReportBCSMEvent	ResetTimer
SendChargingInformation	

Call Associated/Call Processing

Connect	ConnectToResource
EstablishTemporaryConnection	InitiateCallAttempt
CollectInformation	

Procedures at invoking entity (SCF)
A) Sending operation

precondition:	SCSM	any state in which the above operations can be transferred
postcondition:	SCSM	any state as result of the transfer of any of the above operations

B) SCF receives error "MissingParameter"

precondition:	SCSM	any state as result of the transfer of any of the above operations
postcondition:	SCSM	transition to the initial state (i.e. before sending the erroneous operation)

The SL and maintenance functions are informed. Further treatment of the call is dependent on SL.

Procedures at responding entity (SSF)

precondition:	(1) SSF FSM	appropriate state.
	(2) SSF FSM	operation received, appropriate event occurred.
postcondition:	(1) SSF FSM	transition to the same state.

The SSF FSM detects the error in the received operation. The error parameter is returned to inform the SCF of this situation.

8.1.7.3 Operations SSF->SCF

AssistRequestInstructions
InitialDP
ApplyChargingReport

Procedures at invoking entity (SSF)

A) Sending operation

precondition:	SSF FSM	any state in which the above operations can be transferred
postcondition:	SSF FSM	any state as result of the transfer of any of the above operations

B) SSF receives error "MissingParameter"

precondition:	SSF FSM	any state as result of the transfer of any of the above operations.
postcondition:	SSF FSM state a	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.

Procedures at responding entity (SCF)

precondition:	(1) SCSM		appropriate state.
	(2) SCSM		operation received, appropriate event occurred.
postcondition:	(1) SCSM state 1		Idle; in case of InitialDP or ApplyChargingReport
or	(2) SCSM state 2.1		Preparing SSF Instructions; in case of AssistRequestInstructions.

The SCSM detects the erroneous situation. The error parameter is used to inform the SSF of this situation. The SL and maintenance functions are informed.

8.1.7.4 Operations SCF->SRF

PlayAnnouncement
PromptAndCollectUserInformation

Procedures at invoking entity (SCF)

A) Sending operation

precondition:	SCSM state 3.1		Determine Mode; P&C or PA will accompany the ConnectToResource
or	SCSM state 3.2		Waiting for AssistRequestInstructions; after EstablishTemporaryConnection
or	SCSM state 4.1		Waiting for Response from the SRF; if more PAs or P&Cs are outstanding.
postcondition:	SCSM state 4.1		Waiting for Response from the SRF

B) Receiving error

precondition:	SCSM state 4.1	Waiting for Response from the SRF
postcondition:	SCSM state 4.1	Waiting for Response from the SRF

Error treatment depends on SL. SCF can initiate new User Interaction or force Disconnect (to SSF).

Procedures at responding entity (SRF)

precondition:	SRSM state 2	Connected
or	SRSM state 3	User Interaction
postcondition:	SRSM state 3	User Interaction

The SRSM detects that a required parameter is not present in the operation argument. The error parameter MissingParameter is used to inform the SCF of this situation. The SCF should take the appropriate actions to treat this error.

8.1.7.5 Operations SRF->SCF

AssistRequestInstructions

Procedures at invoking entity (SRF)
A) Sending operation
precondition: SRSM state 2 Connected
postcondition: SRSM state 2 Connected

B) Receiving error
precondition: SRSM state 2 Connected
postcondition: SRSM state 1 Idle

Procedures at responding entity (SCF)
precondition: SCSM state 3.2 Waiting for ARI
postcondition: SCSM state 2.1 Preparing SSF instructions
The SCSM detects the error in the received operation. The error parameter is used to inform the SRF of this situation. The SL and maintenance functions are informed. The SCF might try another SRF, route the call or release the call (SL dependent).

8.1.8 ParameterOutOfRange

8.1.8.1 General description

8.1.8.1.1 Error description

The responding entity cannot start the processing of the requested operation because an error in a parameter of the operation argument is detected: a parameter value is out of range.

8.1.8.2 Operations SCF->SSF

Non-call Associated
ActivateServiceFiltering

Call Associated/Non-call Processing
ApplyCharging
CallInformationRequest
SendChargingInformation

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.8.3 Operations SSF->SCF

ApplyChargingReport

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.9 RequestedInfoError

8.1.9.1 General description

8.1.9.1.1 Error description

The RequestedInfoError is an immediate response to the CallInformationRequest operation, indicating that the requested information is not known to the SSF or is not available.

8.1.9.1.2 Argument description

```
PARAMETER ENUMERATED {
        unknownRequestedInfo(1),
        requestedInfoNotAvailable(2)
        }
```

8.1.9.2 Operations SCF->SSF

CallInformationRequest

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.10 SystemFailure

8.1.10.1 General description

8.1.10.1.1 Error description

This error is returned by a PE if it was not able to fulfil a specific task as requested by an operation, and recovery is not expected to be completed within the current call instance.

8.1.10.1.2 Argument description

```
PARAMETER
        UnavailableNetworkResource

UnavailableNetworkResource ::= ENUMERATED {
        unavailableResources(0),
        componentFailure(1),
        basicCallProcessingException(2),
        resourceStatusFailure(3),
        endUserFailure(4)
}
```

8.1.10.2 Operations SCF->SSF

Non-call Associated
ActivateServiceFiltering

Call Associated/Non-call Processing

ApplyCharging	CallInformationRequest
RequestNotificationChargingEvent	RequestReportBCSMEvent
SendChargingInformation	

Call Associated/Call Processing

CollectInformation	Connect
ConnectToResource	DisconnectForwardConnection
EstablishTemporaryConnection	InitiateCallAttempt

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.10.3 Operations SSF->SCF

InitialDP
ApplyChargingReport

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.10.4 Operations SCF->SRF

PlayAnnouncement
PromptAndCollectUserInformation

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.11 TaskRefused

8.1.11.1 General introduction

8.1.11.1.1 Error description

This error is returned by a PE if it was not able to fulfil a specific task as requested by an operation, and recovery is expected to be completed within the current call instance.

8.1.11.1.2 Argument description

```
PARAMETER ENUMERATED {
        generic(0),
        unobtainable(1),
        congestion(2)
        }
```

8.1.11.2 Operations SCF->SSF

Non-call Associated
ActivateServiceFiltering

Call Associated/Non-call Processing

ApplyCharging	CallInformationRequest
FurnishChargingInformation	RequestNotificationChargingEvent
RequestReportBCSMEvent	ResetTimer
SendChargingInformation	

Call Associated/Call Processing

CollectInformation	Connect
ConnectToResource	DisconnectForwardConnection
EstablishTemporaryConnection	InitiateCallAttempt

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.11.3 Operations SSF->SCF

AssistRequestInstructions
InitialDP
ApplyChargingReport

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.11.4 Operations SCF->SRF

PromptAndCollectUserInformation

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.11.5 Operations SRF->SCF

AssistRequestInstructions

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.12 UnavailableResource

8.1.12.1 General description

8.1.12.1.1 Error description

The SRF is not able to perform its function (i.e. play a certain announcement and/or collect specific user information), and cannot be replaced. A reattempt is not possible.

8.1.12.2 Operations SCF->SRF

PlayAnnouncement
PromptAndCollectUserInformation

Procedures at invoking entity (SCF)
A) SCF sends PA or P&C to SRF

precondition:		SCSM state 3.1	Determine Mode; PA or P&C will accompany the ConnectToResource
	or	SCSM state 3.2	Waiting for AssistRequestInstructions; after EstablishTemporaryConnection
	or	SCSM state 4.1	Waiting for Response from the SRF; if more PAs or P&Cs are outstanding.
postcondition:		SCSM state 4.1	Waiting for Response from the SRF

B) SCF receives UnavailableResource error from SRF

precondition:	SCSM state 4.1	Waiting for Response from the SRF
postcondition:	SCSM state 4.1	Waiting for Response from the SRF

If the chosen resource cannot perform its function the further treatment is service dependent.

EXAMPLE:
- request SSF to connect to alternative SRF;
- service processing without PA or P&C (if possible);
- terminate service processing.

Procedures at responding entity (SRF)
A) SRF receiving PA or P&C

precondition:		SRSM state 2	Connected; if initial PA or P&C
	or	SRSM state 3	User Interaction; if not initial PA or P&C

B) SRF is not able to perform its function (and cannot be replaced). SRF sends UnavailableResource.

precondition:	SRSM state 3	User Interaction
postcondition:	SRSM state 3	User Interaction

8.1.13 UnexpectedComponentSequence

8.1.13.1 General description

8.1.13.1.1 Error description

The responding entity cannot start the processing of the requested operation because a SACF or MACF rule is violated, or the operation could not be processed in the current state of the FSM.

8.1.13.2 Operations SCF->SSF

Non-call Associated
ActivateServiceFiltering

Call Associated/Non-call Processing

ApplyCharging	CallInformationRequest
FurnishChargingInformation	RequestNotificationChargingEvent
RequestReportBCSMEvent	ResetTimer
SendChargingInformation	

Call Associated/Call Processing

CollectInformation	Connect
ConnectToResource	Continue
DisconnectForwardConnection	EstablishTemporaryConnection
InitiateCallAttempt	

In this case the SSF detects the erroneous situation, sends the UnexpectedComponentSequence error and remains in the same state. In the SCF the SL and maintenance functions are informed and the SL decides about error treatment.

8.1.13.3 Operations SSF->SCF

ApplyChargingReport
AssistRequestInstructions
InitialDP

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. In that case the SCF returns the error parameter. SL and maintenance are informed. On receiving the error the assisting SSF moves to Idle and the temporary connection is released.

In case the operation is sent by an "initiating" SSF in the context of an existing relationship, the SCF returns the error parameter. SL and maintenance are informed. On receiving the error the SSF moves to Idle.

8.1.13.4 Operations SCF->SRF (only applicable for direct SCF-SRF case)

PlayAnnouncement
PromptAndCollectUserInformation

In this case the SRF detects the erroneous situation, sends the UnexpectedComponentSequence error and remains in the same state. In the SCF the SL and maintenance functions are informed and the SL decides about error treatment. Possible error treatment is to send the DisconnectForwardConnection operation to the SSF.

8.1.13.5 Operations SRF->SCF

AssistRequestInstructions

In this case an error occurs if the SRF has already an established relationship with the SCF and for some reason sends a AssistRequestInstructions. The SCF detects the erroneous situation, informs SL and maintenance functions and returns the error parameter. On receiving the parameter the SRF moves to idle and releases the temporary connection.

8.1.14 UnexpectedDataValue

8.1.14.1 General description

8.1.14.1.1 Error description

The responding entity cannot complete the processing of the requested operation because a parameter has an unexpected data value.

NOTE: This error does not overlap with "ParameterOutOfRange".

EXAMPLE: startTime DateAndTime ::= -- value indicating January 32 1993, 12:15:01

The responding entity does not expect this value and responds with "UnexpectedDataValue".

8.1.14.2 Operations SCF->SSF

Call Associated/Non-call Processing

ApplyCharging
FurnishChargingInformation
RequestNotificationChargingEvent
RequestReportBCSMEvent
ResetTimer

Call Associated/Call Processing

CollectInformation
Connect
ConnectToResource
EstablishTemporaryConnection
InitiateCallAttempt

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.14.3 Operations SSF->SCF

AssistRequestInstructions
InitialDP
ApplyChargingReport

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.14.4 Operations SCF->SRF

PlayAnnouncement
PromptAndCollectUserInformation

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.14.5 Operations SRF->SCF

AssistRequestInstructions

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.15 UnexpectedParameter

8.1.15.1 General description

8.1.15.1.1 Error description

There is an error in the received operation argument. A valid but unexpected parameter was present in the operation argument. The presence of this parameter is not consistent with the presence of the other parameters. The responding entity cannot start to process the operation.

8.1.15.2 Operations SCF->SSF

Non-call Associated

ActivateServiceFiltering

Call Associated/Non-call Processing

ApplyCharging
CallInformationRequest
FurnishChargingInformation
RequestNotificationChargingEvent
RequestReportBCSMEvent
ResetTimer
SendChargingInformation

Call Associated/Call Processing

CollectInformation	Connect
ConnectToResource	EstablishTemporaryConnection
InitiateCallAttempt	

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.15.3 Operations SSF->SCF

AssistRequestInstructions
InitialDP
ApplyChargingReport

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.15.4 Operations SCF->SRF

PlayAnnouncement
PromptAndCollectUserInformation

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.15.5 Operations SRF->SCF

AssistRequestInstructions

Refer to subclause 8.1.7 for the appropriate error procedures.

8.1.16 UnknownLegID

8.1.16.1 General description

8.1.16.1.1 Error description

This error is used to indicate to the SCF that a specific leg, indicated by the LegID parameter value in the operation, is unknown to the SSF.

8.1.16.2 Operations SCF->SSF

Call Associated/Non-call Processing
SendChargingInformation

Refer to subclause 8.1.7 for the appropriate error procedures.

8.2 Entity related error procedures

The following subclauses 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 10.

8.2.1 Expiration of T_{SSF}

8.2.1.1 General description

8.2.1.1.1 Error description

A timeout occurred in the SSF on the response from the SCF.

8.2.1.2 Procedures SSF->SCF

Procedure at the invoking entity (SSF)
Timeout occurs in SSF on T_{SSF}
precondition: SSF FSM state c Waiting for instructions
or SSF FSM state d Waiting for end of User Interaction
or SSF FSM state e Waiting for end of Temporary connection
postcondition: SSF FSM state a Idle
The SSF FSM aborts the dialogue and moves to the Idle state, the CCF routes the call if necessary (e.g. default routing to a terminating announcement). The abort is reported to the maintenance functions.

Procedure 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 abort is received on a SSF dialogue. The SCF releases all resources related to the dialogue, reports the abort to the maintenance functions and returns to state preparing SSF instructions, if the abort is received on an assisting SSF dialogue.

8.2.2 Expiration of T_{SRF}

8.2.2.1 General description

8.2.2.1.1 Error description

A timeout occurred in the SRF on the response from the SCF. This procedure concerns only the direct SCF-SRF case.

8.2.2.2 Procedures SRF->SCF

Procedure at the invoking entity (SRF)
Timeout occurs in SRF on T_{SRF}
precondition: SRSM state 2 Connected
or SRSM state 3 User Interaction
postcondition: SRSM state 1 Idle
The SRF aborts the dialogue and moves to the Idle state, all allocated resources are de-allocated. The abort is reported to the maintenance functions.

Procedure at the responding entity (SCF)
SCF receives a dialogue abort
precondition: SCSM state 4 User Interaction
postcondition: SCSM state 2 Preparing SSF instructions
The SCF releases all resources related to the dialogue, reports the abort to the maintenance functions and returns to state preparing SSF instructions.

9 Detailed operation procedures

9.1 ActivateServiceFiltering procedure

9.1.1 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.

9.1.1.1 Parameters

- 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.

  - sFBillingChargingCharacteristics:
    This parameter determines the charging to be applied for service filtering. Its content is network specific.

  - informationToSend:
    This parameter indicates an announcement, a tone or display information to be sent to the calling party. At the end of information sending, the call shall be released.

    - inbandInfo:
      This parameter specifies the inband information to be sent.

      - messageID:
        This parameter indicates the message(s) to be sent, it can be one of the following:

        - elementaryMessageID:
          This parameter indicates a single announcement.

        - text:
          This parameter indicates a text to be sent. The text shall be transformed to inband information (speech). The attributes of text may consist of items such as language.

        - elementaryMessageIDs:
          This parameter specifies a sequence of announcements.

        - variableMessage:
          This parameter specifies an announcement with one or more variable parts.

      - numberOfRepetitions:
        This parameter indicates the maximum number of times the message shall be sent to the end-user.

      - duration:
        This parameter indicates the maximum time duration in seconds that the message shall be played/repeated. ZERO indicates endless repetition.

      - interval:
        This parameter indicates the time interval in seconds between repetitions, i.e. the time between the end of the announcement and the start of the next repetition. This parameter can only be used when the number of repetitions is greater than one.

- tone:
  This parameter specifies a tone to be sent to the end-user.

  - toneID:
    This parameter indicates the tone to be sent.

  - duration:
    This parameter indicates the time duration in seconds of the tone to be sent. ZERO indicates infinite duration.

- displayInformation:
  This parameter indicates a text string to be sent to the end-user. This information can not be received by a PSTN end-user.

- 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".

- releaseCause:
  This parameter provides the cause value used for call release after the "informationToSend" (e.g. announcement) has been sent to the calling party. If "releaseCause" is not present, the default value is the same as the ISUP value decimal 31 (normal unspecified).

- filteringCharacteristics:
  This parameter indicates the severity of the filtering and the point in time when the ServiceFilteringResponse shall be sent. It determines whether the "interval" or the "numberOfCalls" are used.

  - interval:
    After expiration of the interval timer the next call to arrive causes following actions:
    - sending of an InitialDP;
    - sending of a ServiceFilteringResponse;
    - starting again the interval timer.

    When filtering is started the first interval is started.
    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 operation or a ServiceFilteringResponse operation.
    Other values indicate duration in seconds.

  - numberOfCalls:
    The nth call causes an InitialDP and a 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 indicates that none of the calls matching the filtering criteria will result in sending of an InitialDP operation and a ServiceFilteringResponse operation.

- filteringTimeOut:
  This parameter indicates the duration of the filtering. When the time expires, a ServiceFilteringResponse is sent to the SCF and service filtering is stopped. Two approaches are supported ("duration" or "stopTime"):

  - duration:
    If the duration time expires, then service filtering is stopped and the final report is sent to the SCF.
    A duration of 0 indicates that service filtering is to be removed.
    A duration of -1 indicates an infinite duration.
    A duration of -2 indicates a network specific duration.
    Other values indicate duration in seconds.

  - stopTime:
    When the "stopTime" is met then service filtering is stopped and the final report is sent to the SCF. 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.

- filteringCriteria:
  This parameter specifies which calls are filtered based on "serviceKey", "callingAddressValue", "calledAddressValue" or "locationNumber". It is a choice of "servicekey" or "addressAndService".

  - serviceKey:
    This parameter identifies unambiguously the requested IN service for which filtering should be applied.

  - addressAndService:
    This parameter identifies the IN service and dialled number for which filtering should be applied. The geographical area may also be identified ("callingAddressValue" and/or "locationNumber").

    - serviceKey:

    - calledAddressValue:
      This parameter contains the dialled number towards which filtering shall be applied. The complete called party number shall be specified.

    - callingAddressValue:
      This parameter contains the calling party number which identifies the calling party or geographical origin of the call for which filtering shall be applied.

    - locationNumber:
      This parameter identifies the geographical area from which the call to be filtered originates. It is used when "callingAddressValue" does not contain any information about the geographical location of the calling party.

- startTime:
  This parameter defines when filtering is started. If "startTime" is not provided or was already met, the SSF starts filtering immediately.

9.1.2 Invoking entity (SCF)

9.1.2.1 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 SSF Service Filtering Response".

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 "filteringCriteria". The second parameter set replaces the first one.

9.1.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.1.3 Responding entity (SSF)

9.1.3.1 Normal procedure

SSF precondition:
None.

SSF postcondition:
(1) The SSME FSM is in the state "Non-call Associated Treatment".

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 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.

The 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).

9.1.3.2 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 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 is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.2 ActivityTest procedure

9.2.1 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 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.

9.2.1.1 Parameters

None.

9.2.2 Invoking entity (SCF)

9.2.2.1 Normal procedure

SCF preconditions:
(1) A relationship exists between the SCF and the SSF.
(2) The activity test timer expires, $T_{ActTest}$, after which the ActivityTest operation is sent to the SSF.

SCF postcondition:
(1) If a return result "ActivityTest" is received, the SCME resets the activity test timer and takes no further action.

9.2.2.2 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 SSF 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".

9.2.3 Responding entity (SSF)

9.2.3.1 Normal procedure

SSF precondition:
(1) A relationship exists between the SCF and the SSF.

SSF postconditions:
(1) The SSME FSM stays in, or moves to the state "Non-call Associated Treatment".
(2) If the dialogue ID is active and if there is a SSF FSM using the dialogue, the SSME sends a return result "ActivityTest" to the SCF. If there are no other management activities, the SSME FSM returns to the state "Idle Management", or
If the dialogue ID is not active, the TCAP in the SSP will issue a P-Abort, the SSME will in that case never receive the ActivityTest operation and thus will not be able to reply.

9.2.3.2 Error handling

Not applicable.

9.3 ApplyCharging procedure

9.3.1 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.

As several connection configurations may be established during a call, a possibility exists for the ApplyCharging to be invoked at the beginning of each connection configuration, for each party.

The charging scenarios supported by this operation are scenarios 4.1 and 4.2 (refer to Annex B).

9.3.1.1 Parameters

- 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 contents is network operator specific.

- sendCalculationToSCPIndication:
  This parameter indicates that ApplyChargingReport operations (at least one at the end of the connection configuration charging process) are expected from the SSF. This parameter is always set to TRUE.

- partyToCharge:
  This parameter indicates the party in the call to which the ApplyCharging operation should be applied. If it is not present, then it is applied to the A-party.

9.3.2 Invoking entity (SCF)

9.3.2.1 Normal procedure

SCF Preconditions
(1) A control relationship exists between the SCF and the SSF.
(2) The SLPI has determined that an ApplyCharging operation has to be sent.

SCF postconditions:
(1) No FSM state transition.
(2) The SLPI is expecting ApplyChargingReport operations from the SSF.

The SCSM FSM is in state "Preparing SSF Instructions" or is in state "Queuing FSM". 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.

9.3.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services used for reporting operation errors are described in Clause 10.

9.3.3 Responding entity (SSF)

9.3.3.1 Normal procedure

SSF preconditions:
(1) The SSF FSM is in one of the following states:
"Waiting for Instructions" (state c); or
"Waiting for End of User Interaction" (state d); or
"Waiting for End of Temporary Connection" (state e); or
the assisting/hand-off SSF FSM is in state: "Waiting for Instructions" (state b)

SSF postcondition:
(1) No FSM state transition

On receipt of this operation, the SSF sets the charging data using the information elements included in the operation and acts accordingly. In addition, the SSF will start the monitoring of the end of the connection configuration and other charging events, if requested.

9.3.3.2 Error handling

MissingParameter: This error is indicated if the "sendCalculationToSCPIndication" is not provided.

UnexpectedDataValue: This error is indicated if the "sendCalculationToSCPIndication" is set to FALSE.

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services used for reporting operation errors are described in Clause 10.

9.4 ApplyChargingReport procedure

9.4.1 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. At least one ApplyChargingReport operation shall be sent at the end of the connection configuration charging process.

The charging scenarios supported by this operation are scenarios 4.1 and 4.2 (refer to Annex B).

9.4.1.1 Parameters

- CallResult:
  This parameter provides the SCF with the charging related information previously requested using the ApplyCharging operation with its "sendCalculationToSCPIndication" parameter set to TRUE. The "CallResult" will include the "partyToCharge" parameter as received in the related ApplyCharging operation to correlate the result to the request. The remaining content of "CallResult" is network operator specific. Examples of these contents may be: bulk counter values, costs, tariff change and time of change, time stamps, durations, etc.

9.4.2 Invoking entity (SSF)

9.4.2.1 Normal procedure

SSF preconditions:
(1) A control 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 connection configuration does not change then no FSM state transition shall occur.
  If the connection configuration changes then the FSM shall move to:
  - "Idle" state if there is no other EDP armed and no report requests are pending; or otherwise
  - 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. Examples of charging events may be: threshold value reached, timer expiration, tariff change, end of connection configuration, etc.

9.4.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services used for reporting operation errors are described in Clause 10.

9.4.3 Responding entity (SCF)

9.4.3.1 Normal procedure

SCF preconditions:
(1) An ApplyCharging operation with its "sendCalculationToSCPIndication" parameter set to TRUE has been sent at the request of an SLPI and the SLPI is expecting an ApplyChargingReport from the SSF.

SCF postconditions:
(1) No FSM 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 Call-InformationReport is expected.

On receipt of this operation, the SLPI which is expecting this operation will continue.

9.4.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services used for reporting operation errors are described in Clause 10.

9.5 AssistRequestInstructions procedure

9.5.1 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.

9.5.1.1 Parameters

- correlationID:
  This parameter is used by the SCF to associate the AssistRequestInstructions from the assisting SSF or by a SRF with the InitialDP from the initiating SSF. The value of the "correlationID" may be extracted from the digits received from the initiating SSF or be all of the digits.

- iPAvailable:
  This parameter is sent by the assisting or hand-off SSP to indicate whether or not an IP is attached and available (i.e. not exhausted) at the SSP. This parameter is applicable to this operation only in the physical scenarios corresponding to assist with relay or hand-off. The use of this parameter is network operator dependent.

- iPSSPCapabilities:
  This parameter is sent by the assisting or hand-off SSP to indicate which SRF resources are supported within the SSP and attached and available. This parameter is applicable to this operation only in the physical scenarios corresponding to assist with relay or hand-off. The use of this parameter is network operator dependent.

9.5.2 Invoking entity (SSF/SRF)

9.5.2.1 Normal procedure

SSF precondition:
(1) An assist indication is detected by the assisting SSF.

SSF postcondition:
(1) The assisting SSF waits 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. The AssistRequestInstructions operation is invoked by the SSF or SRF after the call, which initiated the assist indication, is accepted. The assisting SSF FSM transitions to state "Waiting For Instructions".

9.5.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.5.3 Responding entity (SCF)

9.5.3.1 Normal procedure

SCF preconditions:
(1) A control relationship exists between the SCF and the initiating SSF.
(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 PlayAnnouncement or PromptAndCollect-UserInformation 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 PlayAnnouncement or PromptAndCollectUserInformation to be sent to the SRF.

9.5.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.6 CallGap procedure

9.6.1 General description

This operation is used to request the SSF to reduce the rate at which specific service requests are sent to the SCF.

9.6.1.1 Parameters

- gapCriteria:
  This parameter identifies the criteria for a call to be subject to call gapping.
  - calledAddressValue:
    This parameter indicates that call gapping will be applied when the leading digits of the dialled number of a call attempt match those specified in "gapCriteria".
  - gapOnService:
    This parameter indicates that call gapping will be applied when the "servicekey" of a call attempt match those specified in "gapCriteria".
  - calledAddressAndService:
    This parameter indicates that call gapping will be applied when the "serviceKey" and the leading digits of the dialled number of a call attempt match those specified in "gapCriteria".
  - callingAddressAndService:
    This parameter indicates that call 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".
- gapIndicators:
  This parameter indicates the gapping characteristics.
  - duration:
    Duration specifies the total time interval during which call gapping for the specified gap criteria will be active.
    A duration of 0 indicates that gapping is to be removed.
    A duration of -1 indicates an infinite duration.
    A duration of -2 indicates a network specific duration.
    Other values indicate duration in seconds.

- gapInterval:
  This parameter specifies the minimum time between calls being allowed through.
  An interval of 0 indicates that calls meeting the gap criteria are not to be rejected.
  An interval of -1 indicates that all calls meeting the gap criteria are to be rejected.
  Other values indicate interval in milliseconds.

- controlType:
  This parameter indicates the reason for activating call gapping.
  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 "manuallyInitiated" will have priority over "sCPOverloaded" call gap.
  It should be noted that also non-IN controlled traffic control mechanism can apply to an exchange with the SSF functionality. As the non-IN controlled traffic control is within the CCF, this traffic control has implicit priority over the IN controlled traffic control. The non-IN controlled traffic control may also have some influence to the IN call. Therefore it is recommended to take measures to co-ordinate several traffic control mechanisms. The non-IN controlled traffic control and co-ordination of several traffic control mechanisms are out of the scope of core INAP.

- gapTreatment:
  This parameter indicates how calls that were stopped by the call gapping mechanism shall be treated.

  - informationToSend:
    This parameter indicates an announcement, a tone or display information to be sent to the calling party. At the end of information sending, the call shall be released.

    - inbandInfo:
      This parameter specifies the inband information to be sent.

      - messageID:
        This parameter indicates the message(s) to be sent, it can be one of the following:

        - elementaryMessageID:
          This parameter indicates a single announcement.

        - text:
          This parameter indicates a text to be sent. The text shall be transformed to inband information (speech). The attributes of text may consist of items such as language.

        - elementaryMessageIDs:
          This parameter specifies a sequence of announcements.

        - variableMessage:
          This parameter specifies an announcement with one or more variable parts.

      - numberOfRepetitions:
        This parameter indicates the maximum number of times the message shall be sent to the end-user.

      - duration:
        This parameter indicates the maximum time duration in seconds that the message shall be played/repeated. ZERO indicates endless repetition.

- interval:
  This parameter indicates the time interval in seconds between repetitions, i.e. the time between the end of the announcement and the start of the next repetition. This parameter can only be used when the number of repetitions is greater than one.

- tone:
  This parameter specifies a tone to be sent to the end-user.

  - toneID:
    This parameter indicates the tone to be sent.

  - duration:
    This parameter indicates the time duration in seconds of the tone to be sent. ZERO indicates infinite duration.

- displayInformation:
  This parameter indicates a text string to be sent to the end-user. This information can not be received by a PSTN end-user.

- releaseCause:
  This parameter indicates that the call shall be released using the given release cause.

- both:
  This parameter indicates inband information, a tone or display information to be sent to the calling party. At the end of information sending, the call shall be released, using the given release cause.

9.6.2 Invoking entity (SCF)

9.6.2.1 Normal procedure

SCF preconditions:
(1) The SCF detects an overload condition persists and call gapping has to be initiated at the SSF; or
The SCF receives a manually initiated call gapping request from the SMF.

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.

9.6.2.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.6.3 Responding entity (SSF)

9.6.3.1 Normal procedure

SSF preconditions:
(1) Call gapping for gapCriteria is not active; or
Call gapping for gapCriteria is active.

SSF postconditions:
(1) The SSME FSM is in the state "Non-call associated treatment".
(2) Call gapping for gapCriteria is activated; or
Call 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 then a new SSME FSM is created. This SSME FSM enters the state "Non-call associated treatment" and initializes call gapping for the specified IN calls. The parameters "gapIndicators", "controlType" and "gapTreatment" for the indicated gap criteria will be set as provided by the CallGap operation.

In general, 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 will check if call 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 call gapping was initiated for "calledAddressAndService" or "callingAddressAndService" and the "serviceKey" matches, a check on the "calledAddressValue" and "callingAddressValue" (and optionally "locationNumber") for active call 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 to a controlled called number matches several active "gapCriteria", then only the "gapCriteria" associated with the longest called 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. Furthermore, if both "manuallyInitiated" and "sCPOverloaded" "controlType" values are active for this "gapCriteria", then the "manuallyInitiated" control will be applied.

If call gapping shall be applied and there is no gap interval active, an InitialDP operation can be sent including the "cGEncountered" parameter according to the specified controlType. A new gap interval will be initiated as indicated by "gapInterval".

If a gap interval is active, no InitialDP is sent and the call is treated as indicated by "gapTreatment".

The call gap process is stopped if the indicated duration equals ZERO.

If call gapping proceeds then the SSME FSM remains in the state "Non-call associated treatment". Otherwise, the SSME FSM moves to state "Idle Management".

9.6.3.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.7 CallInformationReport procedure

9.7.1 General description

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 operation.

9.7.1.1 Parameters

- requestedInformationList:
  According to the requested information the SSF sends the appropriate types and values to the SCF.

9.7.2 Invoking entity (SSF)

9.7.2.1 Normal procedure

SSF preconditions:
(1) At least one party disconnects from a call.
(2) Requested call information has been collected.
(3) CallInformationReport is pending due to a previously received CallInformationRequest operation.
(4) A control relationship exists between the SCF and the SSF.

SSF postcondition:
(1) The SSF FSM shall move to the "Idle" state in the case where no other report requests are pending and no EDPs are armed otherwise the SSF FSM shall remain in the same state.

If the SSF FSM executes a state transition caused by one of the following events:

- A party release;
- A party abandon;
- B party release;
- B party busy;
- SSF no answer timer expiration;
- route select failure indicated by the network;
- release call initiated by the SCF,

and a CallInformationRequest is pending then the CallInformationReport operation is sent to the SCF.

If a CallInformationReport has been sent to the SCF then no CallInformationReport is pending, i.e. a further CallInformationReport, e.g. in the case of follow-on, has to be explicitly requested by the SCF.

If the event causing the CallInformationReport is also detected by an armed EDP-R then immediately after CallInformationReport the corresponding EventReportBCSM has to be sent.

If the event causing the CallInformationReport is also detected by an armed EDP-N then immediately before CallInformationReport the corresponding EventReportBCSM has to be sent.

9.7.2.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.7.3 Responding entity (SCF)

9.7.3.1 Normal procedure

SCF preconditions:
(1) An SLPI is expecting CallInformationReport.
(2) A control relationship exists between the SCF and the SSF.

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 SLP 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 SL processing has been completed, then the SCSM moves to the "Idle" state.

When the SCF receives the CallInformationReport operation and the SL processing has not been completed yet, then the SCSM remains in the same state (EventReportBCSM or ApplyChargingReport pending).

9.7.3.2 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.

Operation related error handling is not applicable, due to class 4 operation.

9.8 CallInformationRequest procedure

9.8.1 General description

This operation is used to request the SSF to record specific information about a single call and report it to the SCF using the CallInformationReport operation.

9.8.1.1 Parameters

- requestedInformationTypeList:
  This parameter specifies a list of specific items of information which is requested.
  The list may contain:

  - callAttemptElapsedTime:
    This parameter indicates the duration between the end of INAP processing of operations initiating call setup ("Connect") and the received answer indication from the called party side. In case of unsuccessful call setup the network event indicating the unsuccessful call setup stops the measurement of "callAttemptElapsedTime".

  - callStopTime:
    This parameter indicates the time stamp when the connection is released.

  - callConnectedElapsedTime:
    This parameter indicates the duration between the received answer indication from the called party side and the release of the connection.

- calledAddress
  This parameter indicates the incoming called party address that was received by the SSF (i.e., before translation by the SCF).

- releaseCause:
  This parameter indicates the release cause for the call.

Any set of these values can be requested.

9.8.2 Invoking entity (SCF)

9.8.2.1 Normal procedure

SCF preconditions:
(1) A control 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 SLP 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.

9.8.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.8.3 Responding entity (SSF)

9.8.3.1 Normal procedure

SSF preconditions:
(1) Call origination attempt has been initiated.
(2) A control relationship exists between SSF and SCF.

SSF postconditions:
(1) Requested call information is retained by the SSF.
(2) The SSF is waiting for further instructions.

The SSF may receive the CallInformationRequest operation within an existing call associated (CA) dialogue only.

The CallInformationRequest operation is accepted by the SSF FSM only in the state "Waiting for Instructions". The operation does not lead to any transition to another state.

The SSF allocates a record and stores the requested information if already available and prepares the recording of information items that will become available later, e.g. "callStopTimeValue".

9.8.3.2 Error handling

In any other than the "Waiting for Instruction" state the CallInformationRequest operation will be handled as "out of context".

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.9 Cancel procedure

9.9.1 General description

The SCF uses this class 2 operation to request the SRF to cancel a correlated previous operation.
The operation to be deleted can be either a PlayAnnouncement operation or a PromptAndCollectUserInformation operation.

The cancellation of an operation is indicated via a respective error indication, "Cancelled", to the invoking entity of the cancelled PlayAnnouncement or PromptAndCollectUserInformation operation.

The Cancel operation can also be used to cancel all outstanding requests and enable the state machines (SSF/SRF) to go to "Idle". In this case the Cancel operation does not specify any specific operation to be cancelled.

9.9.1.1 Parameters

- invokeID:
  This parameter specifies which operation is to be cancelled.

- allRequests:
  This parameter indicates that all active requests for EventReportBCSM, EventNotificationCharging, ApplyChargingReport and CallInformationReport should be cancelled.

9.9.2 Invoking entity (SCF)

9.9.2.1 Normal procedure

SCF preconditions:
(1) A control relationship exists between the SCF and the SSF/SRF.
(2) An SLPI in the "Waiting for response from SRF" state has determined that a previously requested operation is to be cancelled; or
A SLPI has determined that it is no longer interested in any reports or notifications from the SSF and that the control relationship should be ended.

SCF postcondition:
(1) The SLPI remains in the "Waiting for Response from SRF" state; or
In case all requests are cancelled, the control relationship with the concerned FE (SSF) is ended. If no other relationships persist, the SCSM FSM returns to "Idle" state.

9.9.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.9.3 Responding entity (SRF)

9.9.3.1 Normal procedure

SRF precondition:
(1) A PA/P&C has been received and the SRF is in the "User Interaction" state.

SRF postcondition:
(1) The execution of the PA/P&C has been aborted and the SRF remains in the "User Interaction" state.

9.9.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.9.4 Responding entity (SSF)

9.9.4.1 Normal procedure

SSF precondition:
(1) The SSF FSM is in the states "Waiting for Instructions" or "Monitoring".

SSF postcondition:
(1) All active request for reports and notifications have been cancelled.
(2) In case the SSF FSM was in state "Monitoring" it shall return to idle; or
In case the SSF FSM was in state "Waiting for Instructions" it will remain in that state.
A subsequent call-processing operation will move the SSF FSM state to "Idle". The call, if in active state, is further treated by SSF autonomously as a normal (non-IN) call.

All resources allocated to the dialogue are released.

9.9.4.2 Error handling

Sending of return error on cancel is not applicable in the cancel "allRequests" case.

9.10 CollectInformation procedure

9.10.1 General description

This is a class 2 operation which is used to request the SSF to perform the basic originating call processing actions which will collect destination information from a calling party (it is normally associated with a RequestReportBCSMEvent operation to arm DP2 and to specify the number of digits to be collected).

This operation uses only the resources of the SSF/CCF to collect the information, unlike PromptAndCollectUserInformation, which uses the capabilities of the SRF. It follows that the use of this operation is only appropriate for a call which has not yet left the setup phase.

9.10.1.1 Parameters

None.

9.10.2 Invoking entity (SCF)

9.10.2.1 Normal procedure

SCF precondition:
(1) 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".

9.10.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.10.3 Responding entity (SSF)

9.10.3.1 Normal procedure

SSF precondition:
(1) An InitialDP operation has been invoked.

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 PIC2.

The operation is only valid in the state "Waiting for Instruction" and after having received an operation RequestReportBCSMEvent for DP2. The SSP has to perform the following actions:

- The SSF cancels T_{SSF}.

- When the requisite number of digits (specified when DP2 was armed) has been received, DP2 will be encountered, an EventReportBCSM operation will be invoked, and the SSF FSM will return to the state "Waiting for Instruction".

9.10.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.11 Connect procedure

9.11.1 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.

9.11.1.1 Parameters

- destinationRoutingAddress:
  This parameter contains the called party number towards which the call is to be routed. The encoding of the parameter is defined in ETS 300 356-1 [7]. The "destinationRoutingAddress" may include the "correlationID" and "scfID" if used in the context of a hand-off procedure, but only if "correlationID" and "scfID" are not specified separately.

- correlationID:
  This parameter is used by the SCF to associate the AssistRequestInstructions operation from the assisting SSF with the InitialDP from the initiating SSF. The "correlationID" is used in the context of a hand-off procedure and only if the correlation ID is not embedded in the "destinationRoutingAddress". The network operators has to decide about the actual mapping of this parameter on the used signalling system.

- scfID:
  This parameter indicates the SCF identifier and enables the assisting SSF to identify which SCF the "destinationRoutingAddress" should be sent to. The scfID is used in the context of a hand-off procedure and only if the SCF ID is not embedded in the "destinationRoutingAddress". The network operators has to decide about the actual mapping of this parameter on the used signalling system.

- 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".

- callingPartyNumber:
  This parameter is used to provide an alternative to the "callingPartyNumber" supplied by the network. It may be used for applications such as UPT, where only the SCF can verify the identity of the calling party. The use of this parameter is operator dependent.

- routeList:
  This parameter 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.

- callingPartysCategory:
  This parameter indicates the type of calling party (e.g., operator, pay phone, ordinary subscriber). The use of this parameter in the context of the Connect operation is to be specified by the network operator.

- originalCalledPartyID:
  This parameter carries the dialled digits if the call has met call forwarding on route to the SSP or is forwarded by the SCP. The use of this parameter in the context of the Connect operation is to be specified by the network operator.

- redirectingPartyID:
  This parameter indicates the directory number the call was redirected from. The use of this parameter in the context of the Connect operation is to be specified by the network operator.

- redirectionInformation:
  This parameter contains forwarding related information, such as redirecting counter. The use of this parameter in the context of the Connect operation is to be specified by the network operator.

- alertingPattern:
  This parameter indicates 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 the terminating local exchange for the subscriber.

- serviceInteractionIndicators:
  This parameter contain indicators sent from the SCP to the SSP for control of the network based services at the originating exchange and the destination exchange.

9.11.2 Invoking entity (SCF)

9.11.2.1 Normal procedure

SCF preconditions:
(1) A control relationship exists between the SCF and the SSF.
(2) An SLPI has determined that a Connect has 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 the SCF if the SL 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 shall maintain sufficient information in order to correlate the subsequent AssistRequestInstructions operation (from the assisting SSF or SRF) to the existing SLPI.

9.11.2.2 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 is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.11.3 Responding entity (SSF)

9.11.3.1 Normal procedure

SSF preconditions:
(1) Call origination attempt has been initiated.
(2) Basic call processing has been suspended at a DP.
(3) The SSF waits for instructions.

SSF postcondition:
(1) The SSF performs the call processing actions to route the call to the specified destination.
(2) In the O-BCSM, when only address information is included in the Connect operation, call processing resumes at PIC 3.
(3) In the O-BCSM, when address information and routing information is included in the Connect operation, call processing resumes at PIC 4.

On receipt of this operation in the SSF FSM 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. 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;

- if no EDPs have been armed and no CallInformationReport or ApplyChargingReport has been requested, the FSM goes to state "Idle" (e9). Otherwise, the FSM goes to state "Monitoring" (e11).

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).

9.11.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.12 ConnectToResource procedure

9.12.1 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.

9.12.1.1 Parameters

- resourceAddress:
  This parameter identifies the physical location of the SRF.

  - iPRoutingAddress:
    This parameter indicates the routing address to set up a connection towards the SRF.

  - none:
    This parameter indicates that the call party is to be connected to a predefined SRF.

- serviceInteractionIndicators:
  This parameter contain indicators sent from the SCP to the SSP for control of the network based services at the originating exchange and the destination exchange.

9.12.2 Invoking entity (SCF)

9.12.2.1 Normal procedure

SCF preconditions:
(1) A control relationship exists between the SCF and the SSF.
(2) The SLPI has determined that additional information from the call party is needed.
(3) The call party is not connected to any other party.
(4) The SCSM FSM is in the state "Routing to Resource", sub-state "Determine Mode".
(5) The SLPI has determined that the SRF can be accessed from the SSF.

SCF postconditions:
(1) The SCSM sends out a PlayAnnouncement or PromptAndCollectUserInformation operation accompanying the ConnectToResource.
(2) The SCSM FSM moves to the state "User Interaction".

9.12.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.12.3 Responding entity (SSF)

9.12.3.1 Normal procedure

SSF preconditions:
(1) Basic call processing has been suspended at a DP and a control relationship has been established.
(2) The SSF FSM is in the state "Waiting for Instructions".

SSF postconditions:
(1) The call is switched to the SRF.
(2) A control relationship to the SRF is established.
(3) The SSF FSM moves to the state "Waiting for End of User Interaction". T_{SSF} is set.

NOTE: The successful connection to the SRF causes a state transition in the SRF FSM from "Idle" to "Connected".

9.12.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.13 Continue procedure

9.13.1 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.

9.13.1.1 Parameters

None.

9.13.2 Invoking entity (SCF)

9.13.2.1 Normal procedure

SCF precondition:
(1) SCSM is in the state "Preparing SSF instructions".

SCF postcondition:
(1) SCSM is in the state "Waiting for Notification of Report", in case monitoring was required, or in the state "Idle", in case no monitoring was required.

The SCSM is in state "Preparing SSF instructions". The Continue operation is invoked by a SLPI. This causes a SCSM 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 SCSM transitions to state "Waiting for Notification of Report".

9.13.2.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.13.3 Responding entity (SSF)

9.13.3.1 Normal procedure

SSF preconditions
(1) BCSM: Basic call processing has been suspended at any DP.
(2) SSF FSM is in the state "Waiting for Instructions".

SSF postconditions
(1) BCSM: Basic call processing continues.
(2) SSF FSM is in the state "Monitoring", because at least one EDP was armed, or a CallInformationReport or ApplyChargingReport was requested; or
SSF FSM is in the state "Idle", because no EDPs were armed and neither the CallInformationReport nor the ApplyChargingReport was requested.

The SSF FSM is in state "Waiting for instructions". The SSME receives the Continue operation and relays it to the appropriate SSF FSM. The SSF FSM transitions to state "Idle" in case no EDPs are armed and no outstanding report requests are present. The SSF FSM transits to state "Monitoring" if at least one EDP is armed, or if there is at least one outstanding report request. Basic call processing is resumed.

9.13.3.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.14 DisconnectForwardConnection procedure

9.14.1 General description

This operation is used in the following two cases:

a) 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 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 and PromptAndCollectUser-Information operations;

b) 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.

9.14.1.1 Parameters

None.

9.14.2 Invoking entity (SCF)

9.14.2.1 Normal procedure

SCF preconditions:
(1) A control 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 PE containing the SRF.

In the SCSM FSM state "User Interaction", sub-state "Waiting for Response from the SRF", this operation is invoked by the SCF when the SL 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".

The DisconnectForwardConnection operation contains no parameter since there may be only one SRF connection to one call.

9.14.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.14.3 Responding entity (SSF)

9.14.3.1 Normal procedure

SSF preconditions:
(1) Call origination attempt has been initiated.
(2) Basic call processing has been suspended at a DP.
(3) The initiating SSF is in the state "Waiting for End of User Interaction" or "Waiting for End of Temporary Connection".

SSF postconditions:
(1) The connection to the SRF or assisting SSF is released.
(2) The SSF is waiting for instructions.

The receipt of DisconnectForwardConnection results in disconnecting the assisting SSF or the PE 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 SSF FSM states "Waiting for End of Temporary Connection" or "Waiting for End of User Interaction". On receipt of this operation in these states, the SSP shall perform the following actions:

- the initiating SSF releases the connection to the assisting SSF or the relay SRF;

- the SSF resets T_{SSF};

- the SSF FSM goes to state "Waiting for Instructions" (e8).

The DisconnectForwardConnection operation contains no parameters.

NOTE: The successful disconnection to the SRF causes a state transition in the SRF FSM to "Idle". A current order (PlayAnnouncement or PromptAndCollectUserInformation) is cancelled and any queued order is discarded.

9.14.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.15 EstablishTemporaryConnection procedure

9.15.1 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 PE.

9.15.1.1 Parameters

- assistingSSPIPRoutingAddress:
  This parameter indicates the destination address of the SRF for assist procedure. The "assistingSSPIPRoutingAddress" may contain embedded within it, a "correlationID" and "scfID", but only if "correlationID" and "scfID" are not specified separately.

- correlationID:
  This parameter is used by the SCF to associate the AssistRequestInstructions from the assisting SSF (or the SRF) with the InitialDP from the initiating SSF. The "correlationID" is used only if the correlation ID is not embedded in the "assistingSSPIPRoutingAddress". The network operators has to decide about the actual mapping of this parameter on the used signalling system.

- scfID:
  This parameter indicates the SCF identifier and enables the assisting SSF to identify which SCF the AssistRequestInstructions should be sent to. The "scfID" is used only if the SCF ID is not embedded in the "assistingSSPIPRoutingAddress". The network operators has to decide about the actual mapping of this parameter on the used signalling system.

- serviceInteractionIndicators:
  This parameter contain indicators sent from the SCP to the SSP for control of the network based services at the originating exchange and the destination exchange.

9.15.2 Invoking entity (SCF)

9.15.2.1 Normal procedure

SCF preconditions:
(1) A control relationship exists between the SCF and the SSF.
(2) The SL has determined that a connection is needed between the SSF and SRF or between the SSF and an assisting SSF.
(3) The call party is not connected to any other party.

SCF postcondition:
(1) The SCF is "Waiting for Assist Request Instructions".

In the SCSM FSM state "Routing to Resource", this operation is invoked by the SCF when the SL determines that an assisting SSF or a Direct SCF-SRF relation is needed. The SCSM FSM then transitions to state "Waiting for Assisting Requested Instructions".

9.15.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.15.3 Responding entity (SSF)

9.15.3.1 Normal procedure

SSF preconditions:
(1) Call origination attempt has been initiated.
(2) Basic call processing has been suspended at a DP.
(3) The SSF waits for instructions.
(4) 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.
(2) The SSF waits for end of temporary connection.

On receipt of this operation in the SSF FSM state "Waiting for Instructions", the SSP has to perform the following actions:

- reset the T_{SSF} to T_{ETC};

- route the call to assisting SSF using "assistingSSPIPRoutingAddress";

- the SSF FSM goes to state "Waiting for End of Temporary Connection" (e7).

9.15.3.2 Error handling

Until the connection setup has been accepted 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). 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 is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.16 EventNotificationCharging procedure

9.16.1 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 Annex B, Clause B.4).

As several charging events may occur during a connection configuration a possibility exists for the ENC operation to be invoked on multiple occasions. For each connection configuration ENC may be used several times.

9.16.1.1 Parameters

- eventTypeCharging:
  This parameter indicates the charging event type which has occurred. Its content is network operator specific, which may be "chargePulses" or "chargeMessages".

- eventSpecificInformationCharging:
  This parameter contains charging related information specific to the event. Its content is network operator specific.

- legID:
  This parameter indicates the leg ID on which the charging event type applies.

- monitorMode:
  This parameter indicates how the charging event is reported. 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.

9.16.2 Invoking entity (SSF)

9.16.2.1 Normal procedure

SSF preconditions:
(1) A control 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 SSF FSM 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 by what the charging event is caused 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.

9.16.2.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.16.3 Responding entity (SCF)

9.16.3.1 Normal procedure

SCF precondition:
(1) A RequestNotificationChargingEvent has been sent at the request of a SLPI and 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

9.16.3.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.17 EventReportBCSM procedure

9.17.1 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.

9.17.1.1 Parameters

- eventTypeBCSM:
  This parameter specifies the type of event that is reported.

- eventSpecificInformationBCSM:
  This parameter indicates the call related information specific to the event.
  For CollectedInfo it will contain the "calledPartyNumber".
  For AnalyzedInformation it will contain the "calledPartyNumber".
  For RouteSelectFailure it will contain the "FailureCause", if available.
  For O- or T-CalledPartyBusy it will contain the "BusyCause", if available.
  For O- or T-NoAnswer it will be empty.
  For O- or T-Answer it will be empty.
  For O- or T-MidCall it will be empty.
  For O- or T-Disconnect it will contain the "releaseCause", if available.

- legID:
  This parameter indicates the party in the call for which the event is reported. SSF will use the option "receivingSideID" only.

  - receivingSideID:
    The following values for "legID" are assumed:
    "legID" = 1 indicates the party that was present at the moment of the InitialDP (in case of a midcall trigger, the party causing the trigger), or the party that was created with an InitiateCallAttempt operation.
    "legID" = 2 indicates the party that was created with a Connect operation, or in case of a midcall trigger, the party not causing the trigger.

    If not included, the following defaults are assumed:
    "legID" = 1 for the events CollectedInfo, AnalyzedInformation, O-Abandon and T-Abandon,
    "legID" = 2 for the events RouteSelectFailure, O-CalledPartyBusy, O-NoAnswer, O-Answer, T-CalledPartyBusy, T-NoAnswer and T-Answer.
    The "legID" parameter shall always be included for the events O-MidCall, O-Disconnect, T-MidCall and T-Disconnect.

- miscCallInfo:
  This parameter indicates DP related information.

  - messageType:
    This 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".

9.17.2 Invoking entity (SSF)

9.17.2.1 Normal procedure

SSF preconditions:
(1) The SSF FSM shall be in the state "Monitoring"; or
the SSF FSM may be in state "Waiting for Instructions" if the Disconnect DP is armed and encountered; or
the SSF FSM may be in any state if the Abandon DP is armed and encountered.
(2) The BCSM proceeds to an EDP that is armed.

SSF postconditions:
(1) The SSF FSM stays in the state "Monitoring" if the message type was notification and there are still EDPs armed or a CallInformationReport or ApplyChargingReport requested.
(2) The SSF FSM moves to the state "Idle" if the message type was notification and there are no more EDPs armed, no CallInformationReport or ApplyChargingReport are requested.
(3) The SSF FSM moves to the state "Waiting for Instructions" if the message type was request. Call processing is interrupted.

For certain service features it is necessary that the same O-BCSM instance is reused. Examples are follow-on calls.

The decision to reuse the same O-BCSM instance can only be taken by the SCF after certain armed EDP-Rs are reported.

The considered EDP-Rs are:

- RouteSelectFailure;
- O-CalledPartyBusy;
- O-NoAnswer;
- O-Disconnect.

If a EDP-R is met that causes the release of the related leg all EDPs related to that leg are disarmed and the event is reported via EventReportBCSM. To allow the reuse of the same O-BCSM instance the BCSM has to store all call related signalling parameters (e.g. "callingPartyNumber", "callingPartysCategory") until the BCSM instance is released.

In the case the respective EDP-R is met (see list above with O-Disconnect only for "legID" = 2), the A-leg will be held, the B-leg will be released and the SCF is informed via EventReportBCSM.

If the A-party disconnects the call is released whether or not a DP was armed.

9.17.2.2 Error handling

In case the message type is request, on expiration of T_{SSF} before receiving any operation, the SSF aborts the interaction with the SCF and instructs the CCF to route the call if necessary, e.g. to a final announcement.

Operation related error handling is not applicable, due to class 4 operation.

9.17.3 Responding entity (SCF)

9.17.3.1 Normal procedure

SCF preconditions:
(1) A control relationship exists between the SSF and the SCF.
(2) The SCSM FSM is in the state "Preparing SSF Instructions", sub-state "Waiting for Notification or Request".

SCF postconditions:
(1) The SCSM FSM stays in the sub-state "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 SCSM FSM 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 SCSM FSM moves to the state "Preparing SSF 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.

9.17.3.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.18 FurnishChargingInformation procedure

9.18.1 General description

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. A possibility exists for the Furnish Charging Information (FCI) operation to be invoked on multiple occasions. FCI could be applied at the beginning of the call in order to request to start call record generation. In addition FCI can also be applied at the end of the call or connection configuration (e.g., for follow-on calls). In this case, FCI is used to include charge related information into the call record which was started at the beginning of the call. When additional FCIs are used it is recommended to arm an EDP-R (indicating the end of call or connection configuration) to be able to apply FCI before the termination of the call record generation. The charging scenarios supported by this operation are: scenarios 2.2, 2.3 and 2.4 (refer to Annex B)

9.18.1.1 Parameters

- FCIBillingChargingCharacteristics:
  This parameter indicates billing and/or charging characteristics. Its content is network operator specific. Depending on the applied charging scenario the following information elements can be included (refer to Annex B):
  - complete charging record (scenario 2.2);
  - charge party (scenario 2.3);
  - charge level (scenario 2.3);
  - charge items (scenario 2.3);
  - correlationID (scenario 2.4).

9.18.2 Invoking entity (SCF)

9.18.2.1 Normal procedure

SCF preconditions:
(1) A control 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" or is in state "Queuing FSM". This operation is invoked by the SCF if a SLPI results in the request of creating a call record to the SSF or to include some billing or charging information into the default call 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.

9.18.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.18.3 Responding entity (SSF)

9.18.3.1 Normal procedure

SSF preconditions:
(1) SSF FSM State c, "Waiting for Instructions" or
SSF FSM State d, "Waiting for End of User Interaction" or
SSF FSM State e, "Waiting for End of Temporary Connection" or
Assisting/hand-off SSF FSM State b, "Waiting for Instructions".

SSF postcondition:
(1) No FSM state transition.

On receipt of this operation the SSF performs actions to create the call record according the off-line charging scenario which is applicable using the information elements included in the operation:

- registers the complete call record included in the operation;
- generates and registers a call record according the information (charge party, charge level, charge items);
- include the information received "correlationID" in the default call record which is generated and, registered by default at the SSF.

By means of a parameter at the FurnishChargingInformation operation the SCF can initiate the pulse metering function of the SSF.

In that case the SSF shall generate meter pulses according to the applicable charging level, account and record them.

The SSF records charge related data, e.g. the call duration, begin time stamp or end time stamp. Additionally, the SSF records further data if required.

The charging level can be determined by

a) the SCF; or
b) the SSF; or
c) a succeeding exchange; or
d) the post processing function.

If case a) applies, the charging level is included in the FurnishChargingInformation operation.

If case b) applies, the SSF shall determine the charging level based on the corresponding parameters contained in the operation.

If case c) applies, either the FurnishChargingInformation operation contains the corresponding parameters indicating that the charging level shall be determined in a succeeding exchange or the SSF detects during the determination of the charging level based on the provided parameters that the charging level shall be determined in a succeeding exchange.

The SSF can either account received pulses or convert any charging messages received from the B-side to pulses. In both cases, the accumulated pulses are included when the IN call record is generated or ignored.

9.18.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.19 InitialDP procedure

9.19.1 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.

9.19.1.1 Parameters

- serviceKey:
  This parameter 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).

- calledPartyNumber:
  This parameter contains the number used to identify the called party in the forward direction, e.g. the Called party number of ISUP (see ETS 300 356-1 [7]).

- callingPartyNumber:
  This parameter carries the calling party number to identify the calling party or the origin of the call. The encoding of the parameter is defined in ETS 300 356-1 [7].

- callingPartysCategory:
  Indicates the type of calling party (e.g., operator, pay phone, ordinary subscriber).

- originalCalledPartyID:
  This parameter carries the dialled digits if the call has met call forwarding on the route to the SSP.

- locationNumber:
  This parameter is used to convey the geographical area address for mobility services. 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).

- 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 do not indicate the terminal type. For example, an Integrated Services Private Branch eXchange (ISPBX) will have an ISDN type of access, but the end user terminal behind the ISPBX may be ISDN or non-ISDN.

- bearerCapability:
  This parameter indicates the type of the bearer capability connection to the user:
  - bearerCap:
    This parameter contains the value of the DSS1 Bearer Capability parameter in case the SSF is at local exchange level or the value of the ISUP User Service Information parameter in case the SSF is at transit exchange level.

  The parameter "bearerCapability" shall only be included in the InitialDP operation in case the DSS1 Bearer Capability parameter or the ISUP User Service Information parameter is available at the SSP.

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.

- eventTypeBCSM:
  This parameter indicates the armed BCSM DP event, resulting in the InitialDP operation.

- redirectingPartyID:
  This parameter indicates the directory number the call was redirected from.

- redirectionInformation:
  It contains forwarding related information, such as redirecting counter.

- iPAvailable:
  Indicates whether or not an IP is attached and available (i.e., not exhausted) at the SSP.

- iPSSPCapabilities:
  Indicates which SRF resource are supported within the SSP an attached and available.

- cGEncountered:
  This parameter indicates that the related call has passed call gapping.

- additionalCallingPartyNumber:
  The calling party number provided by the access signalling system of the calling user.

- serviceInteractionIndicators:
  This parameter contain indicators sent from the SSP to the SCP for control of the network based services at the originating exchange and the destination exchange.

- 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 (see ETS 300 403-1 [8]) is used.

9.19.2 Invoking entity (SSF)

9.19.2.1 Normal procedure

SSF preconditions:
(1) Call origination attempt has been initiated.
(2) An event has been detected at a DP.
(3) Call gapping and Signalling System No.7 overload are not in effect for the call, and the call is not to be filtered.

SSF postcondition:
(1) A control relationship has been established and the SSF waits for instructions from the SCF.

Following a trigger detection related to an armed TDP-R in the BCSM caused by a call origination attempt, the SSF checks if call gapping, Signalling System No.7 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.

NOTE: This is for further study.

In some service-specific cases, some parameters shall be available (such as "callingPartyNumber" or "callingPartysCategory"). This shall 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 MF signalling is used, it is possible to request the "callingPartysCategory" from a preceding exchange).

Otherwise, the SSF returns call control to the CCF.

A control 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 from excessive call suspension time.

9.19.2.2 Error handling

If the destination SCF is not accessible then the SSF FSM instructs the CCF to route the call if necessary e.g., to a final announcement.

On expiration of T_{SSF} before receiving any operation, the SSF aborts the interaction with the SCF and instructs the CCF to route the call if necessary e.g., to a final announcement.

If the calling party abandons after the sending of InitialDP, then the SSF aborts the control relationship after the first answer message from the SCF has been received.

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.19.3 Responding entity (SCF)

9.19.3.1 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 control relationship to the related SSF is created. A SLPI is invoked for processing the InitialDP operation based on the "serviceKey" parameter. By means of this control relationship, the SCF may influence the basic call processing in accordance with the SL 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.

9.19.3.2 Error handling

If the InitialDP operation is rejected then the SCSM remains in "Idle". The maintenance function is informed and no SLPI is invoked.

The following error cases are indicated to the SSF:

- MissingCustomerRecord:
  The SCF has not found the appropriate SL corresponding to the provided service key or the appropriate service subscriber related SL.

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.20 InitiateCallAttempt procedure

9.20.1 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. An EDP-R shall be armed on answer and all the call failure events, in order to have the SCF treat this call appropriately when either of these events is encountered.

9.20.1.1 Parameters

- destinationRoutingAddress:
  This parameter contains the called party number towards which the call is to be routed. The encoding of the parameter is defined in ETS 300 356-1 [7].

- callingPartyNumber
  This parameter identifies which number shall be regarded as the calling party for the created call. If this parameter is not sent by the SCF, the SSF may supply a network dependent default value.

- alertingPattern:
  This parameter indicates 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 the terminating local exchange for the subscriber.

- serviceInteractionIndicators:
  This parameter contain indicators sent from the SCP to the SSP for control of the network based services at the originating exchange and the destination exchange.

9.20.2 Invoking entity (SCF)

9.20.2.1 Normal procedure

SCF preconditions:
(1) An SLPI has been invoked, no control relationship exists between SCF and SSF.
(2) An SLPI has determined that an InitiateCallAttempt operation should be sent to the SCF.
(3) The SCSM FSM is in state "Idle".

SCF postconditions:
(1) A control relationship is established between the SCF and SSF.
(2) The SCSM FSM is in state "Preparing SSF Instructions".
(3) SLPI execution continues.

The SCSM FSM moves to state "Preparing SSF Instructions" when the SL invokes this operation. In order to enable the establishment of a control 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 DP3 or DP4, DP5, DP6 and DP7. 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 operation to request the SSF to route the call to the specified destination. The SCSM FSM shall proceed as specified at the procedure for the Continue operation.

The above described procedure shall be part of the establishment of the control relationship, i.e. operations up to and including the Continue operation shall be sent together in the same message to the SSF.

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 then the network no answer timer.

9.20.2.2 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 SCSM FSM moves to the Idle state.

Generic error handling for the operation related errors is described in Clause 8, the TCAP services which are used for reporting operation errors are described in Clause 10.

9.20.3 Responding entity (SSF)

9.20.3.1 Normal procedure

SSF precondition:
None.

SSF postconditions:
(1) A new originating BCSM has been created, call processing is suspended at DP1.
(2) The SSF FSM has moved from "Idle" state to state "Waiting for Instructions".

Upon reception of InitiateCallAttempt, the SSF creates a new originating BCSM and suspends the call processing of this BCSM at DP1. 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.

9.20.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8, the TCAP services which are used for reporting operation errors are described in Clause 10.

9.21 PlayAnnouncement procedure

9.21.1 General description

This operation is used for inband interaction with an analogue user or for interaction with an ISDN user.

9.21.1.1 Parameters

- informationToSend:
  This parameter indicates an announcement, a tone or display information to be sent to the end user by the SRF.

  - inbandInfo:
    This parameter specifies the inband information to be sent.

    - messageID:
      This parameter indicates the message(s) to be sent, this can be one of the following:

      - elementaryMessageID:
        This parameter indicates a single announcement.

      - text:
        This parameter indicates a text to be sent. The text shall be transformed to inband information (speech) by the SRF. The attributes of text may consist of items such as language.

- elementaryMessageIDs:
  This parameter specifies a sequence of announcements.

- variableMessage:
  This specifies an announcement with one or more variable parts.

- numberOfRepetitions:
  This parameter indicates the maximum number of times the message shall be sent to the end-user.

- duration:
  This parameter indicates the maximum time duration in seconds that the message shall be played/repeated. ZERO indicates endless repetition.

- interval:
  This parameter indicates the time interval in seconds between repetitions, i.e. the time between the end of the announcement and the start of the next repetition. This parameter can only be used when the number of repetitions is greater than one.

- tone:
  This parameter specifies a tone to be sent to the end-user.

  - toneID:
    This parameter indicates the tone to be sent.

  - duration:
    This parameter indicates the time duration in seconds of the tone to be sent. ZERO indicates infinite duration.

- displayInformation:
  This parameter indicates a text string to be sent to the end-user. This information can not be received by a PSTN end-user.

NOTE: As the current signalling systems (DSS1/ISUP) do not provide an indication whether or not information can be displayed by the user's terminal, in case of user interaction with an ISDN user two consecutive PlayAnnouncement operations are sent. The first contains the display information, the second contains the inband information to be sent to the user. Since the execution of the display information by the SRF should take a limited amount of time, the inband information will be immediately sent by the SRF to the user, in sequence with the display information.

- disconnectFromIPForbidden:
  This parameter indicates whether or not the SRF should be disconnected from the user when all information has been sent.

- requestAnnouncementComplete:
  This parameter indicates whether or not a SpecializedResourceReport shall be sent to the SCF when all information has been sent.

9.21.2 Invoking entity (SCF)

9.21.2.1 Normal procedure

SCF preconditions:
(1) The SLPI detects that information should be sent to the user.
(2) A connection between the user and a SRF has been established.
(3) The SCSM FSM is in the state "User interaction", sub-state "Waiting for response from the SRF".

SCF postconditions:
(1) If "RequestAnnouncementComplete" was set TRUE, the SCSM will stay in sub-state "Waiting for Response from the SRF" and wait for the SpecializedResourceReport.
(2) If "RequestAnnouncementComplete" was set FALSE and more information needs to be sent ("DisconnectFromIPForbidden" was set to TRUE), the SCSM will stay in sub-state "Waiting for Response from the SRF".
(3) If "RequestAnnouncementComplete" was set FALSE and no more information needs to be sent ("DisconnectFromIPForbidden" was set to FALSE), the SCSM will move to the state "Preparing SSF Instructions".

9.21.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.21.3 Responding entity (SRF)

9.21.3.1 Normal procedure

SRF precondition:
(1) The SRSM FSM is in the state "Connected"; or
in the state "User Interaction" if the SRF received previously an operation from the SCF.

SRF postconditions:
(1) The SRF sends the information to the user as indicated by "informationToSend".
(2) The SRSM FSM moves to the state "User Interaction"; or
remains in the same state.
(3) If all information has been sent and "RequestAnnouncementComplete" was set TRUE, the SRSM sends a SpecializedResourceReport operation to the SCF.
(4) If all information has been sent and "disconnectFromIPForbidden" was set FALSE, the SRSM disconnects the SRF from the user.

The announcement send to the end-user is ended in the following conditions:

- if neither "duration" or "numberOfRepetitions" is specified, then the network specific announcement ending conditions shall apply; or
- if "numberOfRepetitions" is specified, when all repetitions have been sent; or
- if "duration" is specified, when the duration has expired. The announcement is repeated until this condition is met; or
- if "duration" and "numberOfRepetitions" is specified, when one of both conditions is satisfied (whatever comes first).

9.21.3.2 Error handling

If a Cancel operation is received before or during the processing of the operation then the operation is immediately cancelled and the error "Cancelled" is reported to the invoking entity.

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.22 PromptAndCollectUserInformation procedure

9.22.1 General description

This operation is used to interact with a call party in order to collect information.

9.22.1.1 Parameters

- collectedInfo:
  - collectedDigits:
    - minimumNbOfDigits:
      If this parameter is missing, the default value is defined to be 1. The "minimumNbOfDigits" specifies the minimum number of valid digits to be collected.
    - maximumNbOfDigits:
      This parameter should always be present and specifies the maximum number of valid digits to be collected. The following applies:
      "maximumNbOfDigits" ≥ "minimumNbOfDigits".
    - endOfReplyDigit:
      This parameter indicates the digit used to signal the end of input.
      In case the "maximumNbOfDigits" = "minimumNbOfDigits", the "endOfReplyDigit" (could be present but) has no further meaning. This parameter can be one or two digits.

      In case the "maximumNbOfDigits" > "minimumNbOfDigits" the following applies:

      If "endOfReplyDigit" is not present, the end of input is indicated:
      - when the inter-digit timer expires; or
      - when the number of valid digits received equals the "maximumNbOfDigits".

      If "endOfReplyDigit" is present, the end of input is indicated:
      - when the inter-digit timer expires; or
      - when the end of reply digit is received; or
      - when the number of valid digits received equals the "maximumNbOfDigits".

      When the end of input is attained, the collected digits are send from SRF to the SCF. In the case the number of valid digits received is less than the "minimumNbOfDigits" when the inter-digit timer expires or when the end of reply digit is received, the input is specified as being erroneous.
  - cancelDigit:
    If this parameter is present, the cancel digit can be entered by the user to request a possible retry. All digits already received by the SRF are discarded and the same PromptAndCollectUserInformation procedure is performed again, thus e.g. the same announcement to request user information is given to the user and information is collected. This parameter can be one or two digits. If this parameter is not present, the user is not able to request a possible retry.
  - startDigit:
    If this parameter is present, the start digit indicates the start of the valid digits to be collected. The digits that are received by the SRF before this start digit is received, are discarded and are not considered to be valid. This parameter can be one or two digits. If this parameter is not present, all received digits are considered to be valid.

- firstDigitTimeOut:
  If this parameter is present, the first digit should be received by the SRF before the first-digit timer expiration. In case the first digit is not received before first-digit timer expiration, the input is regarded to be erroneous. After receipt of the first valid or invalid input digit, the corresponding first-digit timer is stopped.

  If this parameter is not present, then the SRF uses a default value for the first-digit timer in which the first valid or invalid input digit is received.

  If "startDigit" is present, the first-digit timer is stopped after the start digit is received.

- interDigitTimeOut:
  If this parameter is present any subsequent valid or invalid digit, should be received by the SRF before the inter-digit timer expires. As result the inter-digit timer is reset and restarted.

  In case a subsequent valid or invalid digit is not received before the inter-digit timer expires and the number of received valid digits is less than the "minimumNbOfDigits", the input is regarded to be unsuccessful.

  In case a subsequent valid or invalid digit is not received before the inter-digit timer expires and the number of received valid digits is greater than the "minimumNbOfDigits", and less than or equal to the "maximumNbOfDigits", the input is regarded to be successful.

  If the "interDigitTimeOut" is not present, then the SRF uses a default value for the inter-digit time.

- errorTreatment:
  This optional parameter defines what specific action should be taken by the SRF in the event of error conditions occurring. The default value is stdErrorAndInfo.

- interruptableAnnInd:
  This parameter is optional, where the default value is specified being TRUE.

  If this parameter is TRUE, the announcement is interrupted after the first valid or invalid digit is received by the SRF. If the announcement is interrupted, a possible start-digit timer will not apply anymore. However, if the announcement has not been interrupted, a possible start-digit timer is started after the announcement has been finished.

  If this parameter is present and explicitly set to FALSE, the announcement will not be interrupted after the first digit is received by the SRF. The received digits during the announcement are discarded and considered to be invalid. All other specified parameters ("minimumNbOfDigits", "maximumNbOfDigits", "endOfReplyDigit", etc.) do not apply before the announcement has been finished. The possible start-digit timer is started after the announcement has been finished.

- voiceInformation:
  This parameter is optional, where the default value is specified being FALSE. If the "voiceInformation" parameter is FALSE, all valid or invalid digits are entered by DTMF.

  If this parameter is present and explicitly set to TRUE, calling user is required to provide all valid or invalid information by speech. The SRF will perform voice recognition and translation of the provided information into digits. A possible end of reply digit will also have to be provided by speech.

- voiceBack:
  This parameter is optional, where the default value is specified being FALSE. If the "voiceBack" parameter is FALSE, no voice back information is given by the SRF.

  If this parameter is present and explicitly set to TRUE, the valid input digits received by the SRF will be announced back to the calling user immediately after the end of input is received. The invalid input digits will not be announced back to the calling user. A possible end of reply digit is not voiced back.

- disconnectFromIPForbidden:
  This parameter indicates whether the SRF should initiate disconnection to the SSF/CCF after the interaction has been completed. If the parameter is not present or set to TRUE, the SRF shall not initiate disconnection.

- informationToSend:
  This parameter indicates an announcement, a tone or display information to be sent to the end user by the SRF.

  - inbandInfo:
    This parameter specifies the inband information to be sent.

    - messageID:
      This parameter indicates the message(s) to be sent, this can be one of the following:

      - elementaryMessageID:
        This parameter indicates a single announcement.

      - text:
        This parameter indicates a text to be sent. The text shall be transformed to inband information (speech) by the SRF. The attributes of text may consist of items such as language.

      - elementaryMessageIDs:
        This parameter specifies a sequence of announcements.

      - variableMessage:
        This parameter specifies an announcement with one or more variable parts.

    - numberOfRepetitions:
      This parameter indicates the maximum number of times the message shall be sent to the end-user.

    - duration:
      This parameter indicates the maximum time duration in seconds that the message shall be played/repeated. ZERO indicates endless repetition.

    - interval:
      This parameter indicates the time interval in seconds between repetitions, i.e. the time between the end of the announcement and the start of the next repetition. This parameter can only be used when the number of repetitions is greater than one.

  - tone:
    This parameter specifies a tone to be sent to the end-user.

    - toneID:
      This parameter indicates the tone to be sent.

- duration:
  This parameter indicates the time duration in seconds of the tone to be sent. ZERO indicates infinite duration.

- displayInformation:
  This parameter indicates a text string to be sent to the end-user. This information can not be received by a PSTN end-user.

NOTE: As the current signalling systems (DSS1/ISUP) do not provide an indication whether or not information can be displayed by the user's terminal, in case of user interaction with an ISDN user, the "displayInformation" parameter is not used in the "PromptAndCollectUserInformation" operation. Instead a "PlayAnnouncement" operation containing the "displayInformation" parameter followed by a "PromptAndCollectUserInformation" operation containing inband information are sent to the user. Since the execution of the displayed information by the SRF should take a limited amount of time, the inband information will be immediately sent by the SRF to the user, in sequence with the displayed information.

- digitsResponse:
  This parameter contains the information collected from the end-user.

9.22.2 Invoking entity (SCF)

9.22.2.1 Normal procedure

SCF preconditions:
(1) The SLPI detects that information should be collected from the end-user.
(2) A connection between the end-user and a SRF has been established.
(3) The SCSM FSM is in state "User Interaction", sub-state "Waiting for Response from the SRF".

SCF postconditions:
(1) The collected information is received from the SRF as response to the PromptAndCollectUser-Information operation.
(2) If the "disconnectFromIPForbidden" was set to FALSE, the SCSM FSM will move to the state "Preparing SSF Instructions".
(3) Otherwise the SCSM FSM remains in the same state.

The SLPI may continue execution before the response is received from the PromptAndCollectUser-Information operation, more then one operation may be sent to the SRF before the response is received. The "disconnectFromIPForbidden" parameter may only be set to FALSE if the PromptAndCollectUserInformation operation is the last operation sent to the SRF.

9.22.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8, the TCAP services which are used for reporting operation errors are described in Clause 10.

9.22.3 Responding entity (SRF)

9.22.3.1 Normal procedure

SRF precondition:
(1) The SRSM FSM is in the state "Connected", or in state "User Interaction" if the SRF received previously an operation from the SCF.

SRF postconditions:
(1) The SRF has sent the information to the end-user as indicated by "informationToSend".
(2) The collected information from the end-user is sent to the SCF as return result of the PromptAndCollectUserInformation.
(3) If the "disconnectFromIPForbidden" was set to FALSE, the SRF initiates a bearer channel disconnect to the SSF and the SRSM FSM moves to the state "Idle".
(4) Otherwise the SRSM FSM moves to the state "User Interaction"; or
remains in the same state.

The announcement send to the end-user is ended in the following conditions:

- if neither "duration" or "numberOfRepetitions" is specified, then the network specific announcement ending conditions shall apply; or
- if "numberOfRepetitions" is specified, when all repetitions have been sent; or
- if "duration" is specified, when the duration has expired. The announcement is repeated until this condition is met; or
- if "duration" and "numberOfRepetitions" is specified, when one of both conditions is satisfied (whatever comes first).

The above conditions are overruled if the parameter "interruptableAnnInd" is not set to FALSE and the end-user has responded with a digit during the sending of the announcement. In this case the announcement is ended immediately. The above procedures apply only to inband information and tones send to the end-user, for "displayInformation" the end conditions are met upon sending, i.e. no interruption can occur.

The parameter "errorTreatment" specifies how the SRF shall treat the error "ImproperCallerResponse". The default value "stdErrorAndInfo" means that the error shall be reported to SCF as specified in Clause 8. The value "help" indicates that no error shall be reported to SCF but assistance shall be given to the end-user in form of a network dependent default announcement (which may be dependent on the context, i.e. the sent message). The value "repeatPrompt" indicates that no error shall be reported to the SCF but the prompt shall be repeated to the end-user. The last two procedures shall only be done once per PromptAndCollectUserInformation operation.

9.22.3.2 Error handling

If a Cancel operation is received before or during the processing of the operation then the operation is immediately cancelled and the error "Cancelled" is reported to the invoking entity.

Generic error handling for the operation related errors is described in Clause 8, the TCAP services which are used for reporting operation errors are described in Clause 10.

9.23 ReleaseCall procedure

9.23.1 General description

This operation is used to tear down by the SCF an existing call at any phase of the call for all parties involved in the call. This operation may not be sent to an assisting SSF, except in the case of hand-off procedure.

9.23.1.1 Parameters

- Cause
  A number giving an indication to the SSF about the reason of releasing this specific 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.

9.23.2 Invoking entity (SCF)

9.23.2.1 Normal procedure

SCF precondition:
(1) State 2.1, "Preparing SSF instructions".

SCF postconditions:
(1) State 1, "Idle", if no CallInformationReport has to be received from the SSF. All resources (e.g. queue) related to the call are released by the SCF; or
State 2.3, "Waiting for Notification or Request" if a CallInformationReport still has to be received from the SSF.

9.23.2.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.23.3 Responding entity (SSF)

9.23.3.1 Normal procedure

SSF preconditions:
(1) State C, "Waiting for Instructions"; or
State F, "Monitoring".

SSF postcondition:
(1) "Idle", state a, after sending any outstanding CallInformationReport. Possible armed EDPs are ignored. All connections and resources related to the call are released.

9.23.3.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.24 RequestNotificationChargingEvent procedure

9.24.1 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 SL instance. The operation supports the options to cope with the interactions concerning the charging (refer to Annex B, Clause B.4). As several connection configurations may be established during a call a possibility exists for the RNC operation to be invoked on multiple occasions. For each connection configuration a RNC may be used several times.

9.24.1.1 Parameters

- Sequence of ChargingEvent:
  This parameter contains a list of the charging events and the corresponding monitor types and corresponding legs. For each element in the list the following information elements are included:

  - eventTypeCharging:
    This sub-parameter indicates the charging event type. Its content is network operator specific, which may be "chargePulses" or "chargeMessages".

  - monitorMode:
    This sub-parameter indicates the monitorMode applicable for the corresponding "eventTypeCharging" sub-parameter. Monitor may be "interrupted", "notify" and "continue" or "transparent".

- legID:
  This sub-parameter indicates the leg ID of the corresponding event type charging sub-parameter.

9.24.2 Invoking entity (SCF)

9.24.2.1 Normal procedure

SCF preconditions:
(1) A control 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 is in state "Preparing SSF 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.

9.24.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.24.3 Responding entity (SSF)

9.24.3.1 Normal procedure

SSF preconditions:
(1) SSF FSM State c: "Waiting for Instructions"; or
SSF FSM State d: "Waiting for End of User Interaction"; or
SSF FSM State e: "Waiting for End of Temporary Connection"; or
SSF FSM State f: "Monitoring"; or
Assisting/hand-off SSF FSM State b: "Waiting for Instructions".

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 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 is stop the monitoring of a previously requested charging event.

Requested charging events are monitored until ended by a transparent monitor mode (or in the case of charging events) until the end of the connection configuration.

In the case that multiple RequestNotificationChargingEvent operations are received for the same connection configuration with the same "eventTypeCharging" and "legID", only the last received "monitorMode" will apply.

9.24.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.25 RequestReportBCSMEvent procedure

9.25.1 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.

9.25.1.1 Parameters

- bcsmEvents:
  This parameter specifies the event or events of which a report is requested.
  - eventTypeBCSM:
    This parameter specifies the type of event of which a report is requested. Values origAttemptAuthorized and termAttemptAuthorized are not valid for the RequestReportBCSMEvent operation.
  - monitorMode:
    This parameter indicates how the event should be reported. 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, if the "monitorMode" is "transparent", the event shall not be reported.
  - legID:
    This parameter indicates the party in the call for which the event shall be reported. SCF will use the option "sendingSideID" only.
    - sendingSideID:
      The following values for "legID" are assumed:
      "legID" = 1 indicates the party that was present at the moment of the InitialDP (in case of a mid call trigger, the party causing the trigger), or the party that was created with an InitiateCallAttempt operation.
      "legID" = 2 indicates the party that was created with a "Connect" operation, or in case of a mid call trigger, the party not causing the trigger.
      If not included, the following defaults are assumed:
      "legID" = 1 for the events CollectedInfo, AnalyzedInformation, O-Abandon and T-Abandon,
      "legID" = 2 for the events RouteSelectFailure, O-CalledPartyBusy, O-NoAnswer, O-Answer, T-CalledPartyBusy, T-NoAnswer and T-Answer.
      The "legID" parameter shall always be included for the events O-MidCall, O-Disconnect, T-MidCall and T-Disconnect.
  - dPSpecificCriteria:
    This parameter indicates information specific to the EDP to be armed.

- numberOfDigits:
  This parameter 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.

- applicationTimer:
  This parameter 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 shall be shorter than the network no-answer timer.

9.25.2 Invoking entity (SCF)

9.25.2.1 Normal procedure

SCF preconditions:
(1) A control 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 no CallInformationReport or ApplyChargingReport is pending, the controlrelationship with the concerned SSF is ended. If no other relationship persist, the SCSM shall return to "Idle" state.

9.25.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.25.3 Responding entity (SSF)

9.25.3.1 Normal procedure

SSF precondition:
(1) The SSF FSM is in either the state "Waiting for Instructions" or the state "Monitoring".

SSF postconditions:
(1) The requested EDPs have been armed 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 until the corresponding leg is released.
(3) The SSF FSM remains in the same state.
(4) If all EDPs have been disarmed and no CallInformationReport or ApplyChargingReport has been requested, the SSF FSM moves to the state "Idle".

9.25.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.26 ResetTimer procedure

9.26.1 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.

9.26.1.1 Parameters

- timerValue:
  This parameter specified the value to which the T_{SSF} is to be set.

- timerID:
  This parameter has a default value identifying the T_{SSF} timer.

9.26.2 Invoking entity (SCF)

9.26.2.1 Normal procedure

SCF preconditions:
(1) A control relationship exists between the SCF and the SSF.
(2) An SLPI has determined by the $T_{SCF-SSF}$ 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 reset the $T_{SCF-SSF}$ guard timer.

9.26.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.26.3 Responding entity (SSF)

9.26.3.1 Normal procedure

SSF preconditions:
(1) Call origination attempt has been initiated.
(2) Basic call processing has been suspended at a DP.
(3) The SSF FSM is in the "Waiting for Instruction" state or in the "Waiting for End of User Interaction" state or in the "Waiting for End of Temporary Connection" state.

SSF postconditions:
(1) The T_{SSF} timer has been reset.
(2) The SSF FSM remains in the same state.

9.26.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.27 SendChargingInformation procedure

9.27.1 General description

This operation is used to instruct the SSF on the charging information to be sent by the SSF. The sending of charging information can either be by charge pulses or signalling or internal if SSF is located in the Local Exchange (LE). In the LE, either a charge meter can be updated or a standard call record can be created. A possibility exists for the SCI operation to be invoked on multiple occasions. The charging scenario supported by this operation is: scenario 3.2 (refer to Annex B).

NOTE: The interworking between SSF and PSTN is network operator specific. This operation has many PSTN/IN interactions.

9.27.1.1 Parameters

- sCIBillingChargingCharacteristics:
  This parameter indicates billing and/or charging characteristics. Its content is network operator specific. Depending on the applied charging scenario the following information elements can be included (refer to Annex B):

  - charge level (scenario 3.2);
  - chargePulses;
  - chargeMessages.

- legID:
  This parameter indicates where the charging information shall be sent.

9.27.2 Invoking entity (SCF)

9.27.2.1 Normal procedure

SCF preconditions:
(1) A control 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 send back down the call path.

This causes no SCSM FSM state transition.

9.27.2.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.27.3 Responding entity (SSF)

9.27.3.1 Normal procedure

SSF preconditions:
(1) SSF FSM State c: "Waiting for Instructions"; or
SSF FSM State d: "Waiting for End of User Interaction"; or
SSF FSM State e: "Waiting for End of Temporary Connection"; or
SSF FSM State f: "Monitoring"; or
Assisting/hand-off SSF FSM State b: "Waiting for Instructions".

SSF postcondition:
(1) No 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 if SSF is located in the LE. In the LE, either a charge meter can be updated or a standard call record can be created. The interworking between SSF and PSTN is network operator specific. This operation has much PSTN/IN interactions.

For instance, by sending an operation SendChargingInformation the SCF instructs the SSF to initiate the PSTN/ISDN charging functions according to the given information about the charging level to use.

The charging level can be determined either by one of the following functions:

a) the SCF; or
b) the SSF; or
c) the charging function in a succeeding exchange.

In case of the SCF having determined the charging level, the SendChargingInformation operation contains the charging level to be applied.

In case of the SSF to determine the charging level, the SendChargingInformation operation contains the parameters to determine the charging level.

If the charging level was determined by the IN (SCF or SSF), the SSF provides the charging level to be applied to the PSTN/ISDN charging functions (cases a and b).

In case c), the charging level is determined in a succeeding exchange. The SendChargingInformation operation either contains the corresponding parameters indicating this fact or the SSF detects during trying to determine the charging level based on SCF provided parameters that the charging level shall be determined in a succeeding exchange. Based on already existing PSTN/ISDN capabilities the SSF provides the PSTN/ISDN charging functions with the necessary information and backward charge messages shall be transferred down the call path when allowed by the SCF (generated by a succeeding exchange, e.g. an international gateway).

In the scenario described above charging/billing is performed by means of existing mechanisms of the PSTN/ISDN initiated and controlled by the IN.

That means the determination of the charging method (on-line or off-line) and the items to be charged for shall be done in the basic network, just like the charge generation and the charge registration.

9.27.3.2 Error handling

Generic error handling for the operation related errors is described in Clause 8 and the TCAP services which are used for reporting operation errors are described in Clause 10.

9.28 ServiceFilteringResponse procedure

9.28.1 General description

This operation is used to report the values of counters specified in a previous sent ActivateServiceFiltering operation to the SCF.

9.28.1.1 Parameters

- countersValue:
  The parameter contains the count of calls filtered during the filtering period. It is a list of counter identifications and the related values.

- filteringCriteria:
  This parameter is used to address the concerned SL at the SCF.

9.28.2 Invoking entity (SSF)

9.28.2.1 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 SL at the SCF.

Before ServiceFilteringResponse is sent, it is checked whether call gapping criteria are met. 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 concerned resources are released, i.e. the SSME FSM is removed as well.

9.28.2.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.28.3 Responding entity (SCF)

9.28.3.1 Normal procedure

SCF preconditions:
(1) Service filtering is running.
(2) The SCME is in the state "Waiting for SSF Service Filtering Response".

SCF postcondition:
(1) The SCME forwards the received counter values to the SLPI.

The operation is handled by the Service Filtering FSM part of the 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 SL 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 SCME moves to the state "Service Filtering Idle".

9.28.3.2 Error handling

If the SCME is in the state "Service Filtering Idle" an incoming ServiceFilteringResponse operation is ignored.

9.29 SpecializedResourceReport procedure

9.29.1 General description

This operation is used as the response to a PlayAnnouncement operation when the announcement completed indication is set.

9.29.1.1 Parameters

None.

9.29.2 Invoking entity (SRF)

9.29.2.1 Normal procedure

SRF preconditions:
(1) The SRSM FSM is in the state "User Interaction".
(2) A PlayAnnouncement operation is being executed for which the parameter "RequestAnnouncementComplete" was set TRUE.
(3) All information has been sent to the user.

SRF postconditions:
(1) The SRSM FSM remains in the same state.
(2) If the "DisconnectFromIPForbidden" parameter was set FALSE, the SRSM initiates a bearer channel disconnect sequence to the SSF using the applicable bearer channel signalling system after sending the SpecializedResourceReport operation to the SCF. The SRSM FSM moves to the state "Idle".

9.29.2.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

9.29.3 Responding entity (SCF)

9.29.3.1 Normal procedure

SCF precondition:
(1) The SCSM FSM is in the state "User Interaction", sub-state "Waiting for response from the SRF".

SCF postconditions:
(1) The SCSM FSM remains in the same state.
(2) If the SpecializedResourceReport relates to a PlayAnnouncement operation with permission of SRF initiated disconnection, the SCSM FSM moves to the state "Preparing SSF Instructions".

9.29.3.2 Error handling

Operation related error handling is not applicable, due to class 4 operation.

10 Services assumed from TCAP

10.1 Normal procedures

This subclause describes the procedures and TCAP primitives that shall be used for transmitting messages between SSF, SCF, SRF, and SDF under normal operation.

The INAP, as TC-user, uses only the structured dialogue facility provided by TCAP. The following situations can occur when a message is sent between two physical entities:

- a dialogue shall be established: the TC-user issues a TC-BEGIN request primitive;

- a dialogue shall be maintained: the TC-user issues a TC-CONTINUE request primitive;

- a dialogue shall no longer be maintained: the TC-user issues a TC-END request primitive with either basic end or with pre-arranged end depending on the following conditions:

    - basic end:
        - operations leading to a termination of the control relationship can be transmitted by the SCF with a TC-END request primitive (basic) in case the SCF is not interested in the reception of any ERROR or REJECT components for these sent operations;
          once the SCF dialogue resources have been released any ERROR or REJECT components received for these sent operations will be discarded by TC as described in ETS 300 287 [5] (ITU-T Recommendation Q.774);
        - if the SCF entity has received an operation leading to the termination of the control relationship, a TC-END request primitive (basic) with zero components can be sent from the SCF;

    - pre-arranged end:
      in case of an entity being interested in possible ERROR or REJECT messages in response to sent operations leading to a termination of the control relationship, the dialogue is ended with a TC-END request primitive (pre-arranged end) after the last associated operation timer expires. The receiving entity shall end the dialogue with a TC-END request primitive (basic or pre-arranged end) after successful processing of these operations (i.e. the control relationship is terminated);

- a dialogue shall not be established: for class 2 or 4 operations only the sending TC-user issues a TC-BEGIN request primitive and ends the dialogue locally after operation timeout by means of a prearranged end. Upon reception of the TC-BEGIN indication primitive the receiving TC-user shall end the dialogue locally.

10.1.1 SSF-to-SCF messages

10.1.1.1 SSF FSM related messages

A dialogue shall be established when the SSF FSM moves from the state **Trigger Processing** to the state **Waiting for Instructions**. The relevant INAP operation, which can only be the InitialDP operation, shall be transmitted in the same message.

For all other operations sent from the SSF FSM, the dialogue shall be maintained.

The dialogue shall no longer be maintained when the prearranged end condition is met in the SSF. When the SSF FSM makes a state transition to the state **Idle**, the dialogue is locally ended by means of a TC-END request primitive with prearranged end.

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.

10.1.1.2 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.

The dialogue shall no longer be maintained when the prearranged end condition is met in the SSF. When the SSF FSM makes a state transition to the state **Idle**, the dialogue is locally ended by means of a TC-END request primitive with prearranged end.

10.1.1.3 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.

10.1.2 SCF-to-SSF messages

10.1.2.1 SCSM FSM related messages

A dialogue shall be established when the SCSM FSM moves from state **Idle** to state **Preparing SSF Instructions** and an InitiateCallAttempt operation is sent to the SSF.

For subsequent operations sent from the SCSM FSM, the dialogue shall be maintained, 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 either an InitialDP 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 control relationship, by means of a TC-END request primitive (basic end) is possible.

10.1.2.2 SCME FSM related messages

The operations sent from the SCME FSM shall be issued according to the following procedures:

- the dialogue shall be maintained when the ActivityTest operation 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 FSM 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.

10.1.3 SCF-to/from-SRF messages

A dialogue is established when the SRF sends an AssistRequestInstructions operation to the SCF. For all other operations sent from the SRF, the dialogue shall be maintained.

The dialogue shall no longer be maintained when the prearranged end condition is met in both the SRF and SCF. When the SRSM FSM makes a state transition to the state **Idle**, the dialogue is locally ended by means of a TC-END request primitive with prearranged end.

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 SCF can send the last operation to the SRF with a TC-END request primitive (basic end) when the SCF does not expect any further messages and is no longer interested in any possible error and reject messages.

In the relay case, the SRF-SCF relationship uses the SSF-SCF TCAP dialogue. This is possible, because begin and end of the SRF-SCF relationship are embedded in the SSF-SCF relationship. SRF-SCF information shall be exchanged with TC-CONTINUE request primitives.

10.2 Abnormal procedures

This subclause describes the procedures and TCAP primitives that shall be used for reporting abnormal situations between SSF, SCF, SRF, and SDF. The error cases are defined in Clause 8.

The following primitives shall be used to report abnormal situations:

- operation errors, as defined in the core INAP, are reported with TC-U-ERROR request primitive;

- rejection of a TCAP component by the TC-user shall be reported with TC-U-REJECT request primitive;

- a dialogue shall be aborted by the TC-user with a TC-U-ABORT request primitive.

For abnormal situations detected by TCAP the same rules shall apply for transmission of TC-R-REJECT indication as for transmission of TC-U-REJECT request and for transmission of TC-P-ABORT indication as for transmission of TC-U-ABORT request primitive.

In error situations prearranged end shall not be used. In case any AE encounters an error situation the peer entity shall be explicitly notified of the error, if possible. If from any entity's point of view the error encountered requires the relationship to be ended, it shall close the dialogue via a TC-END request primitive with basic end or via a TC-U-ABORT request primitive, depending on whether any pending ERROR or REJECT component is to be sent or not.

In case an entity receives a TC-END indication primitive and after all components have been considered, the FSM is not in a state to terminate the control relationship, an appropriate internal error should be provided.

In cases when a dialogue needs to be closed by the initiating entity before its establishment has been completed (before the first TC indication primitive to the TC-BEGIN request primitive has been received from the responding entity), the TC-user shall issue a TC-END request primitive with prearranged end or a TC-U-ABORT request primitive. The result of these primitives will be only local, any subsequent TC indication received for this dialogue will be handled according to the abnormal procedures as specified in ETS 300 287 [5] (ITU-T Recommendation Q.774).

10.2.1 SCF-to-SSF/SRF messages

Considering that both SSF and SRF do not have the logic to recover from error cases detected on the SCF-SSF/SRF interface, the following shall apply:

- operation errors and rejection of TCAP components shall be transmitted to the SSF respectively SRF 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 respectively the SRF shall abort the dialogue with a TC-U-ABORT request primitive.

10.2.2 SSF/SRF-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.

If the error processing in the SSF/SRF leads to the case where the SSF/SRF is not able to process further SCF operations while the dialogue is to be maintained, the SSF/SRF aborts the dialogue with a TC-U-ABORT request primitive.

The SSF aborts 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 CallInformationRequest (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.

10.3 Dialogue establishment

The establishment of an INAP dialogue involves two application processes as described in subclause 4.3, one that is the dialogue-initiator and one that is the dialogue-responder.

AC negotiation may not be supported in all physical entities and/or all networks.

This procedure is driven by the following signals:

- a TC-BEGIN request primitive from the dialogue-initiator;
- a TC-BEGIN indication primitive occurring at the responding side;
- the first TC-CONTINUE indication primitive occurring at the initiating side or under specific conditions:
  - a TC-END indication primitive occurring at the initiating side;
  - a TC-U-ABORT indication primitive occurring at the initiating side;
  - a TC-P-ABORT indication primitive occurring at the initiating side.

10.3.1 Sending of a TC-BEGIN request primitive

Before issuing a TC-BEGIN request primitive, SACF shall store the AC-name and if present the user-information parameter.

SACF shall request the invocation of the associated operations using the TC-INVOKE service. See subclause 10.8 for a description of the invocation procedure.

After processing of the last invocation request, SACF shall issue a TC-BEGIN request primitive.

The requesting side SACF then waits for a TC indication primitive and will not issue any other requests, except a TC-U-ABORT request or a TC-END request with the release method parameter set to "pre-arranged release".

If no TC indication primitive is expected because no dialogue is to be established according to the rules as stated in subclauses 10.1 and 10.2, SACF will wait for the last associated TCAP operation timer to expire and issue a TC-END request with the release method parameter set to "pre-arranged release".

10.3.2 Receipt of a TC-BEGIN indication

On receipt of a TC-BEGIN indication primitive, SACF shall:

- analyze the application-context-name included in the primitive and if it is supported, process any other indication primitives received from TC as described in subclause 10.8.

Once all the received primitives have been processed, SACF does not accept any primitive from TC, except a TC-P-ABORT indication.

If no dialogue is to be established according to the rules as stated in subclauses 10.1 and 10.2, SACF will wait for the last indication primitive from TC and issue a TC-END request with the release method parameter set to "pre-arranged release".

If the application-context-name included in the primitive is not supported, issue a TC-U-ABORT request primitive. If an alternative application-context can be offered its name is included in the TC-U-ABORT request primitive.

10.3.3 Receipt of the first TC-CONTINUE indication

On receipt of the first TC-CONTINUE indication primitive for a dialogue, SACF shall check the value of the application-context-name parameter. If this value matches the one used in the TC-BEGIN request primitive, SACF shall process the following TC component handling indication primitives as described in subclause 10.8, otherwise it shall issue a TC-U-ABORT request primitive.

10.3.4 Receipt of a TC-END indication

On receipt of a TC-END indication primitive in the dialogue initiated state, SACF shall check the value of the application-context-name parameter. If this value matches the one used in the TC-BEGIN request primitive then the SACF shall process the following TC component handling indication primitives as described in subclause 10.8, otherwise it shall not be processed.

10.3.5 Receipt of a TC-U-ABORT indication

Receipt of a TC-U-ABORT indication primitive is described as part of user abort procedure (see subclause 10.6.2).

If the abort reason is application-context-name not supported, the responding side may propose an alternative application-context-name in the TC-U-ABORT indication. If an alternative application context is proposed the receiving entity shall check this name and if it can be supported a new dialogue may be established.

10.3.6 Receipt of a TC-P-ABORT indication

Receipt of a TC-P-ABORT indication primitive is described as part of provider abort procedure (see subclause 10.7.1).

10.4 Dialogue continuation

Once established the dialogue is said to be in a continuation phase.

Both application processes can request the transfer of INAP APDUs until one of them requests the termination of the dialogue.

10.4.1 Sending entity

SACF shall process any component handling request primitives as described in subclause 10.8.

After processing the last component handling request primitive, SACF shall issue a TC-CONTINUE request primitive.

10.4.2 Receiving entity

On receipt of a TC-CONTINUE indication primitive SACF shall accept zero, one or several TC component handling indication primitives and process them as described in subclause 10.8.

10.5 Dialogue termination

Both the dialogue-initiator and the dialogue-responder have the ability to request the termination of a dialogue when no dialogue is to be established or when a dialogue is no longer to be maintained according to the rules as stated in subclauses 10.1 and 10.2.

The dialogue termination procedure is driven by the following events:

- a TC-END request primitive;

- a TC-END indication primitive.

10.5.1 Sending of TC-END request

When the dialogue shall no longer be maintained, SACF shall process any component handling request primitives as described in subclause 10.8.

After processing the last component handling request primitive (if any), SACF shall issue a TC-END request primitive with the release method parameter set to "basic end" or "pre-arranged release", according to the rules as stated in subclauses 10.1 and 10.2.

When no dialogue is to be established, refer to subclauses 10.3.1 and 10.3.2.

10.5.2 Receipt of a TC-END indication

On receipt of a TC-END indication primitive, the SACF shall accept any component handling indication primitives and process them as described in subclause 10.8.

After processing the last component handling primitive all dialogue related resources are released.

10.6 User Abort

Both the dialogue-initiator and the dialogue-responder have the ability to abort a dialogue at any time.

The user abort procedure is driven by one of the following events:

- a TC-U-ABORT request primitive;

- a TC-U-ABORT indication primitive.

10.6.1 Sending of TC-U-ABORT request

After issuing a TC-U-ABORT request primitive, all dialogue related resources are released.

10.6.2 Receipt of a TC-U-ABORT indication

On receipt of a TC-U-ABORT indication all dialogue related resources are released.

10.7 Provider Abort

TC has the ability to abort a dialogue at both the dialogue-initiator side and the dialogue-responder side.

The provider abort procedure is driven by the following event:

- a TC-P-ABORT indication primitive.

10.7.1 Receipt of a TC-P-ABORT indication

On receipt of a TC-P-ABORT indication, all dialogue related resources are released.

10.8 Procedures for INAP operations

This subclause describes the procedures for INAP operations.

10.8.1 Operation invocation

SACF shall build an operation argument from the parameters received and request the invocation of the associated operation using the TC-INVOKE procedure. If a linked ID parameter is inserted in the primitive this indicates a child operation and implies that the operation is linked to a parent operation.

10.8.2 Operation invocation receipt

On receipt of a TC-INVOKE indication primitive, SACF shall:

- if the invoke ID is already in use by an active operation, request the transfer of a reject component using the TC-U-REJECT request primitive with the appropriate problem code (duplicated invokeID);

- if the operation code does not correspond to an operation supported by the application-context, request the transfer of a reject component using the TC-U-REJECT request primitive, with the appropriate problem code (unrecognized operation);

- if a linked ID is included, perform the following checks: If the operation referred to by the linked ID does not allow linked operations or if the operation code does not correspond to a permitted linked operation, or if the parent operation invocation is not active, issue a TC-U-REJECT request primitive with the appropriate problem code (linked response unexpected or unexpected linked operation);

- if the type of the argument is not the one defined for the operation, request the transfer of a reject component using the TC-U-REJECT request primitive, with the appropriate problem code (mistyped parameter);

- if the operation cannot be invoked because the dialogue is about to be released, requests the transfer of a reject component using the TC-U-REJECT request primitive with the problem code (Initiating Release);

- if sufficient INAP related resources are not available to perform the requested operation, request the transfer of a reject component using the TC-U-REJECT request primitive with the problem code (Resource Limitation);

- otherwise, accept the TC-INVOKE indication primitive. If the operation is to be user confirmed, SACF waits for the corresponding response.

10.8.3 Operation response

For user confirmed operations, SACF shall:

- if no error indication is included in the response to a class 1 or 3 operation, construct a result information element from the parameters received and request its transfer using the TC-RESULT-L service;

- if an error indication is included in the response to a class 1 or 2 operation, construct an error parameter from the parameters received and request its transfer using the TC-U-ERROR request primitive.

10.8.4 Receipt of a response

10.8.4.1 Receipt of TC-RESULT-NL indication

On receipt of a TC-RESULT-NL indication, SACF shall:

- request the transfer of a reject component using the TC-U-REJECT request primitive, with the appropriate problem code (mistyped parameter).

10.8.4.2 Receipt of TC-RESULT-L indication

On receipt of a TC-RESULT-L indication, SACF shall:

- if the type of the result parameter is not the one defined for the result of this operation, request the transfer of a reject component using the TC-U-REJECT request primitive, with the appropriate problem code (mistyped parameter);

- otherwise, accept the TC-RESULT-L indication primitive.

10.8.4.3 Receipt of TC-U-ERROR indication

On receipt of a TC-U-ERROR indication, SACF shall:

- if the error code is not defined for the SACF or is not one associated with the operation referred to by the invoke identifier, request the transfer of a reject component using the TC-U-REJECT request primitive, with the appropriate problem code (unrecognized error or unexpected error);

- if the type of the error parameter is not the one defined for this error, request the transfer of a reject component using the TC-U-REJECT request primitive, with the appropriate problem code (mistyped parameter);

- otherwise, accept the TC-U-ERROR indication primitive.

10.8.4.4 Receipt of TC-U-REJECT indication

On receipt of a TC-U-REJECT indication primitive which affects a pending operation, SACF shall accept the TC-U-REJECT indication primitive.

10.8.4.5 Receipt of a TC-L-REJECT indication

This event occurs when the local TC detects a protocol error in an incoming component which affects an operation.

On receipt of a TC-L-REJECT indicating "return result problem, return result unexpected", SACF shall inform the application process.

On receipt of a TC-L-REJECT indicating "return error problem, return error unexpected", SACF shall inform the application process.

When the problem code indicates a general problem, it is considered that the event cannot be related to an active operation even if the invoke ID is provided by TC. This is because it is unclear whether the invoke ID refers to a local or remote invocation. The behaviour of SACF in such a case is described in subclause 10.8.5.3.

10.8.4.6 Receipt of a TC-L-CANCEL indication

On receipt of a TC-L-CANCEL indication, the SACF shall:

- if the associated operation is a class 1 operation, inform the application process;
- if the associated operation is a class 2 operation and no linked operations are defined for this operation, ignore the primitive;
- if the associated operation is a class 2 operation and has linked operations but none of them has been invoked, inform the application process;
- if the associated operation is a class 2 operation and a linked operation invocation has already been received in response to this operation, ignore the primitive;
- if the associated operation is a class 3 operation, inform the application process;
- if the associated operation is a class 4 operation, ignore the primitive.

10.8.5 Other events

This subclause describes the behaviour of SACF on receipt of a component handling indication primitive which cannot be related to any operation or which does not affect a pending one.

10.8.5.1 Receipt of a TC-U-REJECT

On receipt of a TC-U-REJECT indication primitive which does not affect an active operation (i.e. indicating a return result or return error problem), 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 in subclause 10.2. This is also applicable for invoke problems related to a class 4 linked operation.

10.8.5.2 Receipt of a TC-R-REJECT indication

On receipt of a TC-R-REJECT indication (i.e. when a protocol error has been detected by the peer TC entity) which does not affect an active operation, 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 in subclause 10.2.

10.8.5.3 Receipt of a TC-L-REJECT indication

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.

10.8.5.4 Receipt of a TC-NOTICE indication

This informs the SACF that a message cannot be delivered by the Network Layer, this can only occur if the Return Option has been set (see subclause 10.9.1.8). It is for the application process to decide whether to terminate the dialogue or retry.

10.9 Mapping on to TC services

10.9.1 Dialogue control

The TC-UNI service is not used by INAP.

10.9.1.1 Destination address

This parameter is set by the dialogue initiating application process, and may optionally be modified by the responding dialogue in the first backward TC-CONTINUE.

10.9.1.2 Originating address

This parameter is set by the dialogue initiating application process.

10.9.1.3 Dialogue ID

The value of this parameter is associated with the INAP invocation in an implementation dependent manner.

10.9.1.4 Application-context-name

The application-context-name parameter is set by SACF as defined in subclause 6.4.

10.9.1.5 User information

This parameter may be used by both initiating and responding application processes.

10.9.1.6 Component present

This parameter is used by SACF as described in ETS 300 287 [5] (ITU-T Recommendation Q.771).

10.9.1.7 Termination

The value of the release method parameter of the TC-END request primitive is set by SACF according to the rules as stated in subclauses 10.1 and 10.2.

10.9.1.8 Quality of service

The quality of service of TC request primitives is set by the SACF to the following value:

- sequencing requested;

- return option, this parameter is set by SACF in an implementation dependent manner.

10.9.2 Operation procedures

10.9.2.1 Invoke ID

This parameter is set by the sending application process.

10.9.2.2 Linked ID

This parameter is set by the sending application process.

10.9.2.3 Dialogue ID

The value of this parameter is associated with the INAP invocation in an implementation dependent manner.

10.9.2.4 Class

The value of this parameter is set by SACF according to the type of the operation to be invoked according to subclause 6.1.

10.9.2.5 Operation

The operation code of a TC-INVOKE request primitive is set by the sending application process as defined in subclause 6.4.

SACF shall set the operation code of the TC-RESULT-L primitive (if required) to the same value as the one received at invocation time.

10.9.2.6 Error

The error parameter of the TC-U-ERROR request primitive is set by the sending application process as defined in subclause 6.4.

10.9.2.7 Parameters

The argument parameter of TC-INVOKE primitives is set by the sending application process as defined in subclauses 6.1 and 6.3.

The result parameter of TC-RESULT-L primitives is set by the sending application process as defined in subclauses 6.1 and 6.3.

The parameter of TC-U-ERROR primitives are set by the sending application process as defined in subclauses 6.2 and 6.3.

10.9.2.8 Time out

The value of this parameter is set by SACF according to the type of operation invoked.

10.9.2.9 Last component

This parameter is used by SACF as described in ETS 300 287 [5] (ITU-T Recommendation Q.771).

10.9.2.10 Problem code

This parameter is used by SACF as described in subclause 10.8.

Annex A (informative): Expanded ASN.1 source of core INAP CS1

This annex contains the expanded ASN.1 source of the core INAP CS1.

```
initialDP  OPERATION
   ARGUMENT
       SEQUENCE {
         serviceKey [0] IMPLICIT INTEGER (0..2147483647),
         calledPartyNumber [2] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         callingPartyNumber [3] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         callingPartysCategory [5] IMPLICIT OCTET STRING (SIZE (1)) OPTIONAL,
         cGEncountered [7] IMPLICIT ENUMERATED {
            manualCGencountered  (1),
            scpOverload  (2)} OPTIONAL,
         iPSSPCapabilities [8] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         iPAvailable [9] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         locationNumber [10] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         originalCalledPartyID [12] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         extensions [15] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL,
         highLayerCompatibility [23] IMPLICIT OCTET STRING (SIZE (2)) OPTIONAL,
         serviceInteractionIndicators [24] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         additionalCallingPartyNumber [25] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         forwardCallIndicators [26] IMPLICIT OCTET STRING (SIZE (2)) OPTIONAL,
         bearerCapability [27] CHOICE {
            bearerCap  [0] IMPLICIT OCTET STRING (SIZE (2..??))} OPTIONAL,
         eventTypeBCSM [28] IMPLICIT ENUMERATED {
            origAttemptAuthorized  (1),
            collectedInfo  (2),
            analyzedInformation  (3),
            routeSelectFailure  (4),
            oCalledPartyBusy  (5),
            oNoAnswer   (6),
            oAnswer     (7),
            oMidCall    (8),
            oDisconnect  (9),
            oAbandon    (10),
            termAttemptAuthorized  (12),
            tCalledPartyBusy  (13),
            tNoAnswer   (14),
            tAnswer     (15),
            tMidCall    (16),
            tDisconnect  (17),
            tAbandon    (18)} OPTIONAL,
         redirectingPartyID [29] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         redirectionInformation [30] IMPLICIT OCTET STRING (SIZE (2)) OPTIONAL}
   ERRORS {
      -- missingCustomerRecord -- localValue 6,
      -- missingParameter -- localValue 7,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 0

assistRequestInstructions OPERATION
   ARGUMENT
       SEQUENCE {
         correlationID [0] IMPLICIT OCTET STRING (SIZE (??..??)),
         iPAvailable [1] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         iPSSPCapabilities [2] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         extensions [3] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
   ERRORS {
      -- missingCustomerRecord -- localValue 6,
      -- missingParameter -- localValue 7,
      -- taskRefused -- localValue 12,
```

```
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 16

establishTemporaryConnection OPERATION
   ARGUMENT
       SEQUENCE {
         assistingSSPIPRoutingAddress [0] IMPLICIT OCTET STRING (SIZE (??..??)),
         correlationID [1] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         scfID      [3] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         extensions [4] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL,
         serviceInteractionIndicators [30] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL}
   ERRORS {
      -- eTCFailed -- localValue 3,
      -- missingParameter -- localValue 7,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 17

disconnectForwardConnection OPERATION
   ERRORS {
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14}
 ::= localValue 18

connectToResource OPERATION
   ARGUMENT
       SEQUENCE {
         resourceAddress CHOICE {
            ipRoutingAddress [0] IMPLICIT OCTET STRING (SIZE (??..??)),
            none       [3] IMPLICIT NULL},
         extensions [4] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL,
         serviceInteractionIndicators [30] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL}
   ERRORS {
      -- missingParameter -- localValue 7,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 19

connect    OPERATION
   ARGUMENT
       SEQUENCE {
         destinationRoutingAddress [0] IMPLICIT SEQUENCE SIZE (1) OF
            OCTET STRING (SIZE (??..??)),
         alertingPattern [1] IMPLICIT OCTET STRING (SIZE (3)) OPTIONAL,
         correlationID [2] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         cutAndPaste [3] IMPLICIT INTEGER (0..22) OPTIONAL,
         originalCalledPartyID [6] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         routeList  [7] IMPLICIT SEQUENCE SIZE (1..3) OF
            OCTET STRING (SIZE (??..??)) OPTIONAL,
         scfID      [8] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         extensions [10] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL,
         serviceInteractionIndicators [26] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
```

```
         callingPartyNumber [27] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         callingPartysCategory [28] IMPLICIT OCTET STRING (SIZE (1)) OPTIONAL,
         redirectingPartyID [29] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         redirectionInformation [30] IMPLICIT OCTET STRING (SIZE (2)) OPTIONAL}
   ERRORS {
      -- missingParameter -- localValue 7,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 20

releaseCall OPERATION
   ARGUMENT
       OCTET STRING (SIZE (2..??))
 ::= localValue 22

requestReportBCSMEvent OPERATION
   ARGUMENT
       SEQUENCE {
         bcsmEvents [0] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               eventTypeBCSM [0] IMPLICIT ENUMERATED {
                  origAttemptAuthorized  (1),
                  collectedInfo  (2),
                  analyzedInformation  (3),
                  routeSelectFailure  (4),
                  oCalledPartyBusy  (5),
                  oNoAnswer   (6),
                  oAnswer     (7),
                  oMidCall    (8),
                  oDisconnect  (9),
                  oAbandon    (10),
                  termAttemptAuthorized  (12),
                  tCalledPartyBusy  (13),
                  tNoAnswer   (14),
                  tAnswer     (15),
                  tMidCall    (16),
                  tDisconnect  (17),
                  tAbandon    (18)},
               monitorMode [1] IMPLICIT ENUMERATED {
                  interrupted  (0),
                  notifyAndContinue  (1),
                  transparent  (2)},
               legID      [2] CHOICE {
                  sendingSideID [0] IMPLICIT OCTET STRING (SIZE (1)),
                  receivingSideID [1] IMPLICIT OCTET STRING (SIZE (1))} OPTIONAL,
               dPSpecificCriteria [30] CHOICE {
                  numberOfDigits [0] IMPLICIT INTEGER (1..255),
                  applicationTimer [1] IMPLICIT INTEGER (0..2047)} OPTIONAL},
         extensions [2] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
   ERRORS {
      -- missingParameter -- localValue 7,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 23

eventReportBCSM OPERATION
   ARGUMENT
       SEQUENCE {
         eventTypeBCSM [0] IMPLICIT ENUMERATED {
            origAttemptAuthorized  (1),
            collectedInfo  (2),
            analyzedInformation  (3),
            routeSelectFailure  (4),
            oCalledPartyBusy  (5),
            oNoAnswer   (6),
            oAnswer     (7),
            oMidCall    (8),
```

```
            oDisconnect  (9),
            oAbandon    (10),
            termAttemptAuthorized  (12),
            tCalledPartyBusy  (13),
            tNoAnswer   (14),
            tAnswer     (15),
            tMidCall    (16),
            tDisconnect  (17),
            tAbandon    (18)},
         eventSpecificInformationBCSM [2] CHOICE {
            collectedInfoSpecificInfo [0] IMPLICIT SEQUENCE {
               calledPartyNumber [0] IMPLICIT OCTET STRING (SIZE (??..??))},
            analyzedInfoSpecificInfo [1] IMPLICIT SEQUENCE {
               calledPartyNumber [0] IMPLICIT OCTET STRING (SIZE (??..??))},
            routeSelectFailureSpecificInfo [2] IMPLICIT SEQUENCE {
               failureCause [0] IMPLICIT OCTET STRING (SIZE (2..??)) OPTIONAL},
            oCalledPartyBusySpecificInfo [3] IMPLICIT SEQUENCE {
               busyCause  [0] IMPLICIT OCTET STRING (SIZE (2..??)) OPTIONAL},
            oNoAnswerSpecificInfo [4] IMPLICIT SEQUENCE {},
            oAnswerSpecificInfo [5] IMPLICIT SEQUENCE {},
            oMidCallSpecificInfo [6] IMPLICIT SEQUENCE {},
            oDisconnectSpecificInfo [7] IMPLICIT SEQUENCE {
               releaseCause [0] IMPLICIT OCTET STRING (SIZE (2..??)) OPTIONAL},
            tCalledPartyBusySpecificInfo [8] IMPLICIT SEQUENCE {
               busyCause  [0] IMPLICIT OCTET STRING (SIZE (2..??)) OPTIONAL},
            tNoAnswerSpecificInfo [9] IMPLICIT SEQUENCE {},
            tAnswerSpecificInfo [10] IMPLICIT SEQUENCE {},
            tMidCallSpecificInfo [11] IMPLICIT SEQUENCE {},
            tDisconnectSpecificInfo [12] IMPLICIT SEQUENCE {
               releaseCause [0] IMPLICIT OCTET STRING (SIZE (2..??)) OPTIONAL}} OPTIONAL,
         legID      [3] CHOICE {
            sendingSideID [0] IMPLICIT OCTET STRING (SIZE (1)),
            receivingSideID [1] IMPLICIT OCTET STRING (SIZE (1))} OPTIONAL,
         miscCallInfo [4] IMPLICIT SEQUENCE {
            messageType [0] IMPLICIT ENUMERATED {
               request     (0),
               notification  (1)}} DEFAULT {
            messageType request    },
         extensions [5] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
 ::= localValue 24

requestNotificationChargingEvent OPERATION
   ARGUMENT
       SEQUENCE SIZE (1..??) OF
         SEQUENCE {
            eventTypeCharging [0] IMPLICIT OCTET STRING (SIZE (??..??)),
            monitorMode [1] IMPLICIT ENUMERATED {
               interrupted  (0),
               notifyAndContinue  (1),
               transparent  (2)},
            legID      [2] CHOICE {
               sendingSideID [0] IMPLICIT OCTET STRING (SIZE (1)),
               receivingSideID [1] IMPLICIT OCTET STRING (SIZE (1))} OPTIONAL}
   ERRORS {
      -- missingParameter -- localValue 7,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 25

eventNotificationCharging OPERATION
   ARGUMENT
       SEQUENCE {
         eventTypeCharging [0] IMPLICIT OCTET STRING (SIZE (??..??)),
         eventSpecificInformationCharging [1] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         legID      [2] CHOICE {
            sendingSideID [0] IMPLICIT OCTET STRING (SIZE (1)),
            receivingSideID [1] IMPLICIT OCTET STRING (SIZE (1))} OPTIONAL,
         extensions [3] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
```

```
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL,
         monitorMode [30] IMPLICIT ENUMERATED {
            interrupted  (0),
            notifyAndContinue  (1),
            transparent  (2)} DEFAULT notifyAndContinue }
 ::= localValue 26

collectInformation OPERATION
   ARGUMENT
       SEQUENCE {
         extensions [4] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
   ERRORS {
      -- missingParameter -- localValue 7,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 27

continue   OPERATION
 ::= localValue 31

initiateCallAttempt OPERATION
   ARGUMENT
       SEQUENCE {
         destinationRoutingAddress [0] IMPLICIT SEQUENCE SIZE (1) OF
            OCTET STRING (SIZE (??..??)),
         alertingPattern [1] IMPLICIT OCTET STRING (SIZE (3)) OPTIONAL,
         extensions [4] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL,
         serviceInteractionIndicators [29] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
         callingPartyNumber [30] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL}
   ERRORS {
      -- missingParameter -- localValue 7,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 32

resetTimer OPERATION
   ARGUMENT
       SEQUENCE {
         timerID    [0] IMPLICIT ENUMERATED {
            tssf        (0)} DEFAULT tssf       ,
         timervalue [1] IMPLICIT INTEGER (0..2147483647),
         extensions [2] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
   ERRORS {
      -- missingParameter -- localValue 7,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 33
```

```
furnishChargingInformation OPERATION
   ARGUMENT
       OCTET STRING (SIZE (??..??))
   ERRORS {
      -- missingParameter -- localValue 7,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 34

applyCharging OPERATION
   ARGUMENT
       SEQUENCE {
         aChBillingChargingCharacteristics [0] IMPLICIT OCTET STRING (SIZE (??..??)),
         sendCalculationToSCPIndication [1] IMPLICIT BOOLEAN DEFAULT FALSE,
         partyToCharge [2] CHOICE {
            sendingSideID [0] IMPLICIT OCTET STRING (SIZE (1)),
            receivingSideID [1] IMPLICIT OCTET STRING (SIZE (1))} OPTIONAL,
         extensions [3] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
   ERRORS {
      -- missingParameter -- localValue 7,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedParameter -- localValue 16,
      -- unexpectedDataValue -- localValue 15,
      -- parameterOutOfRange -- localValue 8,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12}
 ::= localValue 35

applyChargingReport OPERATION
   ARGUMENT
       OCTET STRING (SIZE (??..??))
   ERRORS {
      -- missingParameter -- localValue 7,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedParameter -- localValue 16,
      -- unexpectedDataValue -- localValue 15,
      -- parameterOutOfRange -- localValue 8,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12}
 ::= localValue 36

callGap    OPERATION
   ARGUMENT
       SEQUENCE {
         gapCriteria [0] CHOICE {
            calledAddressValue [0] IMPLICIT OCTET STRING (SIZE (??..??)),
            gapOnService [2] IMPLICIT SEQUENCE {
               serviceKey [0] IMPLICIT INTEGER (0..2147483647)},
            calledAddressAndService [29] IMPLICIT SEQUENCE {
               calledAddressValue [0] IMPLICIT OCTET STRING (SIZE (??..??)),
               serviceKey [1] IMPLICIT INTEGER (0..2147483647)},
            callingAddressAndService [30] IMPLICIT SEQUENCE {
               callingAddressValue [0] IMPLICIT OCTET STRING (SIZE (??..??)),
               serviceKey [1] IMPLICIT INTEGER (0..2147483647),
               locationNumber [2] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL}},
         gapIndicators [1] IMPLICIT SEQUENCE {
            duration   [0] IMPLICIT INTEGER (-2..86400),
            gapInterval [1] IMPLICIT INTEGER (-1..60000)},
         controlType [2] IMPLICIT ENUMERATED {
            sCPOverloaded  (0),
            manuallyInitiated  (1)} OPTIONAL,
         gapTreatment [3] CHOICE {
            informationToSend [0] CHOICE {
               inbandinfo [0] IMPLICIT SEQUENCE {
                  messageID  [0] CHOICE {
                     elementaryMessageID [0] IMPLICIT INTEGER (0..2147483647),
                     text       [1] IMPLICIT SEQUENCE {
                        messageContent [0] IMPLICIT IA5String   (SIZE (??..??)),
                        attributes [1] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL},
                     elementaryMessageIDs [29] IMPLICIT SEQUENCE SIZE (1..??) OF
                        INTEGER (0..2147483647),
```

```
                  variableMessage [30] IMPLICIT SEQUENCE {
                     elementaryMessageID [0] IMPLICIT INTEGER (0..2147483647),
                     variableParts [1] IMPLICIT SEQUENCE SIZE (1..5) OF
                        CHOICE {
                           integer    [0] IMPLICIT INTEGER (0..2147483647),
                           number     [1] IMPLICIT OCTET STRING (SIZE (??..??)),
                           time       [2] IMPLICIT OCTET STRING (SIZE (2)),
                           date       [3] IMPLICIT OCTET STRING (SIZE (3)),
                           price      [4] IMPLICIT OCTET STRING (SIZE (4))}}},
               numberOfRepetitions [1] IMPLICIT INTEGER (1..127) OPTIONAL,
               duration   [2] IMPLICIT INTEGER (0..32767) OPTIONAL,
               interval   [3] IMPLICIT INTEGER (0..32767) OPTIONAL},
            tone       [1] IMPLICIT SEQUENCE {
               toneID     [0] IMPLICIT INTEGER (0..2147483647),
               duration   [1] IMPLICIT INTEGER (0..2147483647) OPTIONAL},
            displayInformation [2] IMPLICIT IA5String   (SIZE (??..??))},
         releaseCause [1] IMPLICIT OCTET STRING (SIZE (2..??)),
         both       [2] IMPLICIT SEQUENCE {
            informationToSend [0] CHOICE {
               inbandinfo [0] IMPLICIT SEQUENCE {
                  messageID  [0] CHOICE {
                     elementaryMessageID [0] IMPLICIT INTEGER (0..2147483647),
                     text       [1] IMPLICIT SEQUENCE {
                        messageContent [0] IMPLICIT IA5String   (SIZE (??..??)),
                        attributes [1] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL},
                     elementaryMessageIDs [29] IMPLICIT SEQUENCE SIZE (1..??) OF
                        INTEGER (0..2147483647),
                     variableMessage [30] IMPLICIT SEQUENCE {
                        elementaryMessageID [0] IMPLICIT INTEGER (0..2147483647),
                        variableParts [1] IMPLICIT SEQUENCE SIZE (1..5) OF
                           CHOICE {
                              integer    [0] IMPLICIT INTEGER (0..2147483647),
                              number     [1] IMPLICIT OCTET STRING (SIZE (??..??)),
                              time       [2] IMPLICIT OCTET STRING (SIZE (2)),
                              date       [3] IMPLICIT OCTET STRING (SIZE (3)),
                              price      [4] IMPLICIT OCTET STRING (SIZE (4))}}},
                     numberOfRepetitions [1] IMPLICIT INTEGER (1..127) OPTIONAL,
                     duration   [2] IMPLICIT INTEGER (0..32767) OPTIONAL,
                     interval   [3] IMPLICIT INTEGER (0..32767) OPTIONAL},
                  tone       [1] IMPLICIT SEQUENCE {
                     toneID     [0] IMPLICIT INTEGER (0..2147483647),
                     duration   [1] IMPLICIT INTEGER (0..2147483647) OPTIONAL},
                  displayInformation [2] IMPLICIT IA5String   (SIZE (??..??))},
               releaseCause [1] IMPLICIT OCTET STRING (SIZE (2..??))}} OPTIONAL,
         extensions [4] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
 ::= localValue 41

activateServiceFiltering OPERATION
   ARGUMENT
       SEQUENCE {
         filteredCallTreatment [0] IMPLICIT SEQUENCE {
            sFBillingChargingCharacteristics [0] IMPLICIT OCTET STRING (SIZE (??..??)),
            informationToSend [1] CHOICE {
               inbandinfo [0] IMPLICIT SEQUENCE {
                  messageID  [0] CHOICE {
                     elementaryMessageID [0] IMPLICIT INTEGER (0..2147483647),
                     text       [1] IMPLICIT SEQUENCE {
                        messageContent [0] IMPLICIT IA5String   (SIZE (??..??)),
                        attributes [1] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL},
                     elementaryMessageIDs [29] IMPLICIT SEQUENCE SIZE (1..??) OF
                        INTEGER (0..2147483647),
                     variableMessage [30] IMPLICIT SEQUENCE {
                        elementaryMessageID [0] IMPLICIT INTEGER (0..2147483647),
                        variableParts [1] IMPLICIT SEQUENCE SIZE (1..5) OF
                           CHOICE {
                              integer    [0] IMPLICIT INTEGER (0..2147483647),
                              number     [1] IMPLICIT OCTET STRING (SIZE (??..??)),
                              time       [2] IMPLICIT OCTET STRING (SIZE (2)),
                              date       [3] IMPLICIT OCTET STRING (SIZE (3)),
                              price      [4] IMPLICIT OCTET STRING (SIZE (4))}}},
                     numberOfRepetitions [1] IMPLICIT INTEGER (1..127) OPTIONAL,
                     duration   [2] IMPLICIT INTEGER (0..32767) OPTIONAL,
                     interval   [3] IMPLICIT INTEGER (0..32767) OPTIONAL},
```

```
               tone       [1] IMPLICIT SEQUENCE {
                  toneID     [0] IMPLICIT INTEGER (0..2147483647),
                  duration   [1] IMPLICIT INTEGER (0..2147483647) OPTIONAL},
               displayInformation [2] IMPLICIT IA5String   (SIZE (??..??))} OPTIONAL,
            maximumNumberOfCounters [2] IMPLICIT INTEGER (1..100) OPTIONAL,
            releaseCause [3] IMPLICIT OCTET STRING (SIZE (2..??)) OPTIONAL},
         filteringCharacteristics [1] CHOICE {
            interval   [0] IMPLICIT INTEGER (-1..32000),
            numberOfCalls [1] IMPLICIT INTEGER (0..2147483647)},
         filteringTimeOut [2] CHOICE {
            duration   [0] IMPLICIT INTEGER (-2..86400),
            stopTime   [1] IMPLICIT OCTET STRING (SIZE (6))},
         filteringCriteria [3] CHOICE {
            serviceKey [2] IMPLICIT INTEGER (0..2147483647),
            addressAndService [30] IMPLICIT SEQUENCE {
               calledAddressValue [0] IMPLICIT OCTET STRING (SIZE (??..??)),
               serviceKey [1] IMPLICIT INTEGER (0..2147483647),
               callingAddressValue [2] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
               locationNumber [3] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL}},
         startTime  [4] IMPLICIT OCTET STRING (SIZE (6)) OPTIONAL,
         extensions [5] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
   ERRORS {
      -- missingParameter -- localValue 7,
      -- parameterOutOfRange -- localValue 8,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 42

serviceFilteringResponse OPERATION
   ARGUMENT
       SEQUENCE {
         countersValue [0] IMPLICIT SEQUENCE SIZE (0..100) OF
            SEQUENCE {
               counterID  [0] IMPLICIT INTEGER (0..99),
               counterValue [1] IMPLICIT INTEGER (0..2147483647)},
         filteringCriteria [1] CHOICE {
            serviceKey [2] IMPLICIT INTEGER (0..2147483647),
            addressAndService [30] IMPLICIT SEQUENCE {
               calledAddressValue [0] IMPLICIT OCTET STRING (SIZE (??..??)),
               serviceKey [1] IMPLICIT INTEGER (0..2147483647),
               callingAddressValue [2] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL,
               locationNumber [3] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL}},
         extensions [2] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
 ::= localValue 43

callInformationReport OPERATION
   ARGUMENT
       SEQUENCE {
         requestedInformationList [0] IMPLICIT SEQUENCE SIZE (1..5) OF
            SEQUENCE {
               requestedInformationType [0] IMPLICIT ENUMERATED {
                  callAttemptElapsedTime  (0),
                  callStopTime  (1),
                  callConnectedElapsedTime  (2),
                  calledAddress  (3),
                  releaseCause  (30)},
               requestedInformationValue [1] CHOICE {
                  callAttemptElapsedTimeValue [0] IMPLICIT INTEGER (0..255),
                  callStopTimeValue [1] IMPLICIT OCTET STRING (SIZE (6)),
                  callConnectedElapsedTimeValue [2] IMPLICIT INTEGER (0..2147483647),
                  calledAddressValue [3] IMPLICIT OCTET STRING (SIZE (??..??)),
                  releaseCauseValue [30] IMPLICIT OCTET STRING (SIZE (2..??))}},
         extensions [2] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
```

```
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
 ::= localValue 44

callInformationRequest OPERATION
   ARGUMENT
       SEQUENCE {
         requestedInformationTypeList [0] IMPLICIT SEQUENCE SIZE (1..5) OF
            ENUMERATED {
               callAttemptElapsedTime  (0),
               callStopTime  (1),
               callConnectedElapsedTime  (2),
               calledAddress  (3),
               releaseCause  (30)},
         extensions [2] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
   ERRORS {
      -- missingParameter -- localValue 7,
      -- parameterOutOfRange -- localValue 8,
      -- requestedInfoError -- localValue 10,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 45

sendChargingInformation OPERATION
   ARGUMENT
       SEQUENCE {
         sCIBillingChargingCharacteristics [0] IMPLICIT OCTET STRING (SIZE (??..??)),
         legID      [1] CHOICE {
            sendingSideID [0] IMPLICIT OCTET STRING (SIZE (1)),
            receivingSideID [1] IMPLICIT OCTET STRING (SIZE (1))},
         extensions [2] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
   ERRORS {
      -- missingParameter -- localValue 7,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedParameter -- localValue 16,
      -- parameterOutOfRange -- localValue 8,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unknownLegID -- localValue 17}
 ::= localValue 46

playAnnouncement OPERATION
   ARGUMENT
       SEQUENCE {
         informationToSend [0] CHOICE {
            inbandinfo [0] IMPLICIT SEQUENCE {
               messageID  [0] CHOICE {
                  elementaryMessageID [0] IMPLICIT INTEGER (0..2147483647),
                  text       [1] IMPLICIT SEQUENCE {
                     messageContent [0] IMPLICIT IA5String   (SIZE (??..??)),
                     attributes [1] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL},
                  elementaryMessageIDs [29] IMPLICIT SEQUENCE SIZE (1..??) OF
                     INTEGER (0..2147483647),
                  variableMessage [30] IMPLICIT SEQUENCE {
                     elementaryMessageID [0] IMPLICIT INTEGER (0..2147483647),
                     variableParts [1] IMPLICIT SEQUENCE SIZE (1..5) OF
                        CHOICE {
                           integer    [0] IMPLICIT INTEGER (0..2147483647),
                           number     [1] IMPLICIT OCTET STRING (SIZE (??..??)),
                           time       [2] IMPLICIT OCTET STRING (SIZE (2)),
                           date       [3] IMPLICIT OCTET STRING (SIZE (3)),
                           price      [4] IMPLICIT OCTET STRING (SIZE (4))}}},
               numberOfRepetitions [1] IMPLICIT INTEGER (1..127) OPTIONAL,
```

```
               duration   [2] IMPLICIT INTEGER (0..32767) OPTIONAL,
               interval   [3] IMPLICIT INTEGER (0..32767) OPTIONAL},
            tone       [1] IMPLICIT SEQUENCE {
               toneID     [0] IMPLICIT INTEGER (0..2147483647),
               duration   [1] IMPLICIT INTEGER (0..2147483647) OPTIONAL},
            displayInformation [2] IMPLICIT IA5String   (SIZE (??..??))},
         disconnectFromIPForbidden [1] IMPLICIT BOOLEAN DEFAULT TRUE,
         requestAnnouncementComplete [2] IMPLICIT BOOLEAN DEFAULT TRUE,
         extensions [3] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
   ERRORS {
      -- cancelled -- localValue 0,
      -- missingParameter -- localValue 7,
      -- systemFailure -- localValue 11,
      -- unavailableResource -- localValue 13,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
   LINKED {
      -- specializedResourceReport -- localValue 49}
 ::= localValue 47

promptAndCollectUserInformation OPERATION
   ARGUMENT
       SEQUENCE {
         collectedInfo [0] CHOICE {
            collectedDigits [0] IMPLICIT SEQUENCE {
               minimumNbOfDigits [0] IMPLICIT INTEGER (1..127) DEFAULT 1,
               maximumNbOfDigits [1] IMPLICIT INTEGER (1..127),
               endOfReplyDigit [2] IMPLICIT OCTET STRING (SIZE (1..2)) OPTIONAL,
               cancelDigit [3] IMPLICIT OCTET STRING (SIZE (1..2)) OPTIONAL,
               startDigit [4] IMPLICIT OCTET STRING (SIZE (1..2)) OPTIONAL,
               firstDigitTimeOut [5] IMPLICIT INTEGER (1..127) OPTIONAL,
               interDigitTimeOut [6] IMPLICIT INTEGER (1..127) OPTIONAL,
               errortreatment [7] IMPLICIT ENUMERATED {
                  stdErrorAndInfo  (0),
                  help        (1),
                  repeatPrompt  (2)} DEFAULT stdErrorAndInfo ,
               interruptableAnnInd [8] IMPLICIT BOOLEAN DEFAULT TRUE,
               voiceInformation [9] IMPLICIT BOOLEAN DEFAULT FALSE,
               voiceBack  [10] IMPLICIT BOOLEAN DEFAULT FALSE}},
         disconnectFromIPForbidden [1] IMPLICIT BOOLEAN DEFAULT TRUE,
         informationToSend [2] CHOICE {
            inbandinfo [0] IMPLICIT SEQUENCE {
               messageID  [0] CHOICE {
                  elementaryMessageID [0] IMPLICIT INTEGER (0..2147483647),
                  text       [1] IMPLICIT SEQUENCE {
                     messageContent [0] IMPLICIT IA5String   (SIZE (??..??)),
                     attributes [1] IMPLICIT OCTET STRING (SIZE (??..??)) OPTIONAL},
                  elementaryMessageIDs [29] IMPLICIT SEQUENCE SIZE (1..??) OF
                     INTEGER (0..2147483647),
                  variableMessage [30] IMPLICIT SEQUENCE {
                     elementaryMessageID [0] IMPLICIT INTEGER (0..2147483647),
                     variableParts [1] IMPLICIT SEQUENCE SIZE (1..5) OF
                        CHOICE {
                           integer    [0] IMPLICIT INTEGER (0..2147483647),
                           number     [1] IMPLICIT OCTET STRING (SIZE (??..??)),
                           time       [2] IMPLICIT OCTET STRING (SIZE (2)),
                           date       [3] IMPLICIT OCTET STRING (SIZE (3)),
                           price      [4] IMPLICIT OCTET STRING (SIZE (4))}}},
               numberOfRepetitions [1] IMPLICIT INTEGER (1..127) OPTIONAL,
               duration   [2] IMPLICIT INTEGER (0..32767) OPTIONAL,
               interval   [3] IMPLICIT INTEGER (0..32767) OPTIONAL},
            tone       [1] IMPLICIT SEQUENCE {
               toneID     [0] IMPLICIT INTEGER (0..2147483647),
               duration   [1] IMPLICIT INTEGER (0..2147483647) OPTIONAL},
            displayInformation [2] IMPLICIT IA5String   (SIZE (??..??))} OPTIONAL,
         extensions [3] IMPLICIT SEQUENCE SIZE (1..??) OF
            SEQUENCE {
               type       INTEGER,
               criticality ENUMERATED {
                  ignore      (0),
                  abort       (1)} DEFAULT ignore     ,
               value      [1] ANY DEFINED BY type       } OPTIONAL}
```

```
   RESULT
       CHOICE {
         digitsResponse [0] IMPLICIT OCTET STRING (SIZE (??..??))}
   ERRORS {
      -- cancelled -- localValue 0,
      -- improperCallerResponse -- localValue 4,
      -- missingParameter -- localValue 7,
      -- systemFailure -- localValue 11,
      -- taskRefused -- localValue 12,
      -- unavailableResource -- localValue 13,
      -- unexpectedComponentSequence -- localValue 14,
      -- unexpectedDataValue -- localValue 15,
      -- unexpectedParameter -- localValue 16}
 ::= localValue 48

specializedResourceReport OPERATION
   ARGUMENT
       NULL
 ::= localValue 49

cancel     OPERATION
   ARGUMENT
       CHOICE {
         invokeID   [0] IMPLICIT INTEGER (-128..127),
         allRequests [1] IMPLICIT NULL}
   ERRORS {
      -- cancelFailed -- localValue 1}
 ::= localValue 53

activityTest OPERATION
   RESULT
       NULL
 ::= localValue 55

cancelled   ERROR
 ::= localValue 0

cancelFailed ERROR
   PARAMETER
       SEQUENCE {
         problem    [0] IMPLICIT ENUMERATED {
            unknownOperation  (0),
            tooLate     (1),
            operationNotCancellable  (2)},
         operation  [1] IMPLICIT INTEGER (-128..127)}
 ::= localValue 1

eTCFailed  ERROR
 ::= localValue 3

improperCallerResponse ERROR
 ::= localValue 4

missingCustomerRecord ERROR
 ::= localValue 6

missingParameter ERROR
 ::= localValue 7

parameterOutOfRange ERROR
 ::= localValue 8

requestedInfoError ERROR
   PARAMETER
       ENUMERATED {
         unknownRequestedInfo  (1),
         requestedInfoNotAvailable  (2)}
 ::= localValue 10

systemFailure ERROR
   PARAMETER
       ENUMERATED {
         unavailableResources  (0),
         componentFailure  (1),
         basicCallProcessingException  (2),
         resourceStatusFailure  (3),
         endUserFailure  (4)}
 ::= localValue 11
```

```
taskRefused ERROR
   PARAMETER
       ENUMERATED {
         generic     (0),
         unobtainable  (1),
         congestion  (2)}
 ::= localValue 12

unavailableResource ERROR
 ::= localValue 13

unexpectedComponentSequence ERROR
 ::= localValue 14

unexpectedDataValue ERROR
 ::= localValue 15

unexpectedParameter ERROR
 ::= localValue 16

unknownLegID ERROR
 ::= localValue 17
```

Annex B (informative): Charging scenarios supported by core INAP

B.1 Introduction

This annex provides information on how the different charging capabilities in INAP CS1 may be used. Networks may support different or additional charging capabilities then listed in this annex.

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, specific charging operations have to be transferred via the INAP. This annex describes the IN charging requirements from an INAP point of view. First some terminology concerning charging processes and charging capabilities is listed, followed by particular charging scenarios for which the INAP requirements are listed. Via the information flows and information elements, the corresponding charging operations for the INAP can be defined.

B.2 Charging requirements

In general two types of charging capabilities are required.

B.2.1 Off-line charging

The usage and/or charge information of the call is recorded in the network. 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).

B.2.2 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.

B.3 Charging processes

Following considerations apply to the case of one service and one network.

At a high level the following processes concerning the charging can be identified:

Charge determination process (DET):
- determining the charge party. Charge party can be the calling line or IN service subscriber or both;
- determining the level of charge;
- determining the items to be charged;
- if determination of the previous elements is performed off-line then in that case only a default call record is registered.

Charge generation process (GEN):
- generation of charge pulses or charge related signalling or charge related information for the off-line process.

Charge registration process (REG):
- updating the charge meters or creating call records or both.

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.

Charge Output Process (OUT):
- output of charging data for further processing. Charging data can be output to magnetic tapes or datalinks, on operator request or scheduled.

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.

B.4 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 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.

For each of the scenarios (and therefore for each charged party) there is only one point of control for IN charging per call.

B.4.1 Charging scenarios related to off-line charging

To support the off-line charging process, some charging related functions have to be executed during a call instance.

B.4.1.1 Scenario 1: IN charging completely in the PSTN

In this case (scenario 1), the charging is done by the existing charging mechanisms in the PSTN, 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.

B.4.1.2 Scenario 2: IN charging completely in the IN

In this scenario, the charging is done completely in the IN nodes. 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 (scenario 2.2, 2.3) or SCF (scenario 2.1) or both (scenario 2.4).

In case call records are registered at both SSF and SCF, for the same call instance there should be a correlation between both call records to allow the off-line billing process to correlate them. For this purpose the SCF should generate a unique correlationID and send it to the SSF. Both the SCF and SSF should register this in the call record.

B.4.1.3 Scenario 3: IN Charging shared between IN and PSTN

In this case, the SCF has control of the charging information and instructs the SSF on the charging information to be sent by the SSF (scenario 3.2).

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.

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 need not know whether the SSP is a transit or a local exchange.

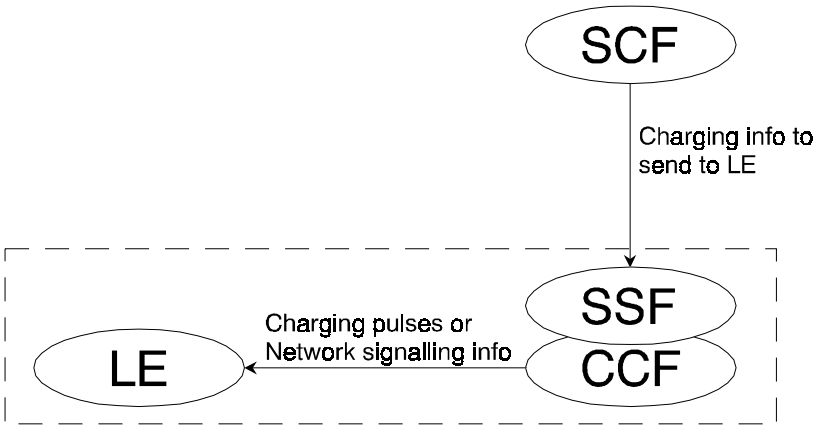


Figure B.1: Scenario 3

B.4.1.4 Scenario 4: Charging at the SCF, assisted by the SSF

The SCF has the relevant charging information to apply charging and instructs the SSF to calculate the call charge. Included in these instructions are the conditions on which the SSF should request further information from the SCF (e.g. thresholds, i.e. inform the SCF when the charge reaches a certain amount).

When the call has finished or e.g. a threshold is reached, then the SSF informs the SCF on what has happened and the SCF performs the necessary processing of the charge information, and possibly instructs the SSF what to do next (e.g. clear the call).

The calculated call charge can be registered at either the SCF (scenario 4.1) or SSF (scenario 4.2).

B.4.2 Charging scenarios related to on-line charging

Apart from the scenarios supporting the off-line charging, some networks require scenarios to support the on-line charging capability. This means that at the user/network interface charge related information (like charge pulses or signalling information) have to be offered.

If the charging is controlled by the SCF, it shall be possible to offer the charge information to the LE. The LE uses this information at the UNI. If this charge information is delivered to the LE for call recording purposes (i.e. scenario 3, scenario 4 in case SSF in LE), this information may also be used for on-line charging.

If this charge information is not delivered to the LE (i.e. scenario 2 and 4 in case SSF not in LE), then additional IFs are needed for on-line charging. In that case the same configuration described in scenario 3 could be used to transfer the charge information in addition to the off-line charging related scenario.

Flexible tariffing is a concept used in many networks. For IN calls, charge rates need to be changed at certain times to all or dedicated destinations, or per call instance dependent on the duration of the conversation or the service interactions involved.

When SCF controls the charging and on-line charging is required for a call instance, the SCF should be able to change the charge rates in the network. This on-line charging process may cause additional signalling traffic and processing load in both the IN and the PSTN/ISDN.

To avoid additional processing and signalling in the network it may be useful to apply the on-line charging selectively (i.e. only on the calls for which it is necessary, e.g. payphone or AOC). Whether on-line charging is required or not can be determined by either:

- an on-line charging indication from PSTN/ISDN to IN; or

- the SL (e.g. based on serviceKey, user service profile or user interaction).

B.5 Interactions

Interactions concerning the charging of an IN call can occur in the following cases:

1) a higher exchange generating charge related signalling towards the SSP.
2) call parties controlled by different SLs.

B.5.1 Interactions with other networks concerning charging control

This is the case when SSF receives charging-related signalling from a higher exchange (e.g. international exchange or dedicated service exchanges).

There are three options identified to handle this type of interaction. For a call instance controlled by the SCF, the SCF should select one of these options. According to the general requirement "there is only one point of control for charging per call party per call", the selected option will remain during the lifetime of the call instance for the call party.

Option 1: In this option SCF has control of the charging information and instructs SSF to monitor and intercept the charge related signalling message(s) (from ISUP and not charge pulses) received from a higher exchange. Based on criteria supplied by the SCF, the SSF will send this information to the SCF:

1) immediately after receipt of the appropriate message type;
2) after receipt of a specified number of messages of the appropriate type; or
3) at the end of the call.

Subsequently 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 to be sent to the SSF for on-line charging).

Option 2: The SCF may also decide to leave the charge control to the higher exchange and monitor the charging events. In this case the SCF instructs SSF to use normal charging processing, e.g. send charge signalling information and/or charge pulses through to the LE and/or update charge meters according to the received charging information from the higher exchange, and to notify the SCF. Based on criteria supplied by the SCF, the SSF will send this information to the SCF:

1) immediately after receipt of the appropriate message type;
2) after receipt of a specified number of messages of the appropriate type; or
3) at the end of the call.

Option 3: Same case as for option 2 except that the SCF does not wish to monitor the charging events.

B.5.2 Call parties controlled by different SLs

This is the case for e.g. UPT to UPT or UPT to VPN calls. Interaction occurs in the following cases:

case 1: SL-B causes charging information from SSF FSM-B towards SSF FSM-A;

case 2: SL-A causes charging information from SSF FSM-A towards SSF FSM-B;

case 3: case 1 and case 2.

B.5.2.1 Case 1: charging information from SSF FSM-B to SSF FSM-A

This interaction occurs when SL-B causes a charge pulse or signalling information from SSF FSM-B to SSF FSM-A.

NOTE: If both SSF FSMs reside in the same SSF, then the principles will be the same, but the SSF FSM-A/SSF FSM-B interface will be internal rather than by network signalling.

There are three options identified to handle this type of interaction. For a call instance controlled by SL-A, the SL-A should select one of these options. According to the general requirement "there is only one point of control for charging per call party per call", the selected option will remain during the lifetime of the call instance for the call party.

Option 1: In this option SL-A has control of the charging information for the A-party and instructs SSF FSM-A to monitor and intercept the charge related messages received from SSF FSM-B and send it to the SL-A. Based on criteria supplied by the SL-A the SSF will send this information to the SL-A:

1) immediately after receipt of the appropriate message type;
2) after receipt of a specified number of messages of the appropriate type; or
3) at the end of the call.

Subsequently, SL-A may use this information performing its charging control (use the received charging information in the call record generation, adjust new charge rates/pulses to be sent to the SSF FSM-A for on-line charging mechanism).

Option 2: The SL-A may also decide to leave the charge control to the SL-B but monitor the charging events. In this case the SL-A instructs SSF FSM-A to use normal charging processing, e.g. send charge signalling information and/or charge pulses through to the LE and/or update charge meters according to the received charging information from the SSF FSM-B, and to notify the SL-A. Based on criteria supplied by the SCF, the SSF will send this information to the SCF:

1) immediately after receipt of the appropriate message type;
2) after receipt of a specified number of messages of the appropriate type; or
3) at the end of the call.

Option 3: Same case as for option 2 except that SL-A does not wish to monitor the charging events.

B.5.2.2 Case 2: charging information from SSF FSM-A to SSF FSM-B

This interaction occurs when SL-A causes a charge pulse or signalling information[2)] from SSF FSM-A to SSF FSM-B. To handle this type of interaction the same options apply as case 1 (i.e. SL-B selects one of the options).

B.5.2.3 Case 3: charging information from SSF FSM-A to SSF FSM-B and reverse

This interaction occurs when both SL-B causes a charge pulse or signalling information[2)] from SSF FSM-B to SSF FSM-A and SL-A causes a charge pulse or signalling information[2)] from SSF FSM-A to SSF FSM-B.

This interaction can be managed by a superposition of the principles given for the cases 1 and 2.

B.6 Framework for the charging operations in INAP

In general:

- the starting point for the INAP work on charging is one service and one network;
- charging scenario(s) are pre-defined per service. The decision to perform DET/GEN/REG in the SSF or SCF can be made on a per call basis by the SCF;
- the processes OFC and OUT are not listed in the framework, as they have no impact on the scenarios defined.

---

2) If both SSF FSMs reside in the same SSF, then the principles will be the same, but the SSF FSM-A/SSF FSM-B interface will be internal rather than by network signalling.

Table B.1: Framework charging related operations in INAP

Reference scenario nr.	Applicable	DET	GEN	REG	ONC (LE)	Information to be transferred by INAP charging operations	INAP charging operations	Parameters
1	Yes	PSTN	PSTN	PSTN	y/n	no	no	no
2.1	Yes	SCF	SCF	SCF	n	no	no	no
2.2	Yes	SCF	SCF	SSF	n	SCF -> SSF charge records	FCI (complete charging record	FCI_BCC
2.3	Yes	SCF	SSF	SSF	n	SCF -> SSF Charged Party/Level/Item	FCI	FCI_BCC
2.4	Yes	SCF	SCF	SSF & SCF	n	SCF -> SSF CorrelationID (NOTE 4)	FCI	FCI_BCC
3.2	Yes	SCF	SSF	PSTN	y/n	SCF -> SSF Charged Party/Level (NOTE 1)	SCI	SCI_BCC legID
4.1 (NOTE 5)	Yes	SCF	SSF	SCF	y/n (NOTE 2)	SCF -> SSF Charged Party/Level/Item SCF <- SSF Call charge or threshold (NOTE 3)	AC ACR	AC_BCC partyToCharge (Optional) sendCalculationToSCP-Indication CallResult
4.2	Yes	SCF	SSF	SSF	y/n	SCF -> SSF Charge Level SCF <- SSF threshold (NOTE 3)	AC FCI (NOTE 6) ACR	AC_BCC partyToCharge (Optional) sendCalculationToSCP-Indication FCI_BCC CallResult

NOTE 1: No elaboration on the interworking of the SSF to the PSTN will be made because the objective is to define charging information transferred by INAP. This scenario has many PSTN/IN interactions.
NOTE 2: On-line charging applicable in case LE SSF is located in the LE otherwise network signalling needed.
NOTE 3: The threshold is not intended for statistics but used by services like credit card calling.
NOTE 4: In this case, SSF creates a default/standard call record and includes a correlationID supplied by SCF. It is up to the network operator to use an additional scenario to perform on-line charging (e.g. scenario 3.2).
NOTE 5: In this scenario, the SCF generates the call record to be used by the post processing centre to determine the party to charge and the cost of the call.
NOTE 6: FurnishChargingInformation controls the record generation at the SSF. ApplyCharging and ApplyChargingReport are used to transfer charging related information from the SSF to the SCF (e.g. debit card calling).

Annex C (informative): Trigger Detection Points (TDPs) and trigger criteria for the CS1 core INAP

C.1 Introduction

This annex provides for the core INAP information on the applicability of the CS1 Detection Points (DPs) as Trigger Detection Points (TDPs) and on the applicability of trigger criteria that are listed and clarified in ITU-T Recommendation Q.1219, § 6.3.3.3.

This annex has no intention to be restrictive on the use of the EventTypeBCSM parameter in the InitialDP operation. All values of the EventTypeBCSM are possible.

In addition the indicated applicability of trigger criteria to the TDPs are considered as being useful in the context of core INAP and it is not intended to be restrictive.

Note, however, that the assignment of DP to a TDP on either a customer/trunkgroup, private facility or office basis may have an impact on the memory and real time performance requirements of the CCF/SSF.

Note further that the applicability of DP criteria at a given DP depends on when call processing is available and how long it is retained. The core INAP places no requirements on equipment suppliers in this area. If network and service providers plan to implement IN CS1 services in a multi-supplier environment, they should consider formulating such requirements to ensure consistent implementations across supplier equipments. Such requirements should be considered carefully so as not to adversely impact memory and real-time performance of CCF/SSF processing.

The relationship between DP criteria and minimum parameter population rules is not defined in this version of the ETS.

C.2 BCSM TDPs and TDP criteria

The following classification is used.

C.2.1 Unconditional TDPs

The unconditional TDPs can be used by itself, related to DPs of the BCSM without any other criterion to be satisfied at the TDP before informing the SCF that the TDP was encountered. The unconditional TDPs can be:

- customer based;
- trunkgroup based;
- private facility (e.g. Centrex) based.

The unconditional TDPs are listed in table C.1.

Table C.1

DP criteria	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18
UNCONDITIONAL Line based	X	X	X	X	X	X	-	-	X	-	X	X	X	-	-	X	-
UNCONDITIONAL Trunkgroup based	X	X	X	X	X	X	-	-	X	-	X	X	X	-	-	X	-
UNCONDITIONAL Private facility based	X	X	X	X	X	X	-	-	X	-	X	X	X	-	-	X	-
Key: X : applicable - : not applicable																	

A "X" for the selected DPs means really the point in call when the BCSM should be suspended. E.g. for DP3 the information should only be analyzed whereafter, irrespective the value of the analyzed information the BCSM should be suspended.

C.2.2 Conditional TDPs

Related to DPs with additional trigger criteria as listed in the table C.2.

The entries in the table can be:

- customer based;
- trunkgroup based;
- private facility based;
- office based.

Table C.2

DP criteria	1	2	3	4	5	6	7	8	9	10	12	13	14	15	16	17	18
Class of service	X	O	O	O	O	O	O	O	O	O	X	O	O	O	O	O	O
Specific digit string (NOTE 1)	-	X	X	O	O	O	O	O	O	O	-	-	-	-	-	-	-
Feature code (NOTE 1)	-	X	X	O	O	O	O	O	O	O	-	-	-	-	-	-	-
Prefixes (NOTE 1)	-	X	X	O	O	O	O	O	O	O	-	-	-	-	-	-	-
Access Codes (NOTE 1)	-	X	X	O	O	O	O	O	O	O	-	-	-	-	-	-	-
Called party number (NOTE 1)	-	X	X	O	O	O	O	O	O	O	-	-	-	-	-	-	-
Facility information (NOTE 2)	-	-	X	-	-	-	X	X	-	-	-	-	-	X	X	-	-
Feature activation (NOTE 3)	-	-	X	X	X	X	X	X	X	X	-	-	X	X	X	X	X
Cause	-	-	-	X	X	-	-	-	X	X	-	X	-	-	-	X	X
Specific ABD string (NOTE 1)	-	-	X	O	O	O	O	O	O	O	-	-	-	-	-	-	-
Specific calling party number (NOTE 4)	X	O	O	O	O	O	O	O	O	O	X	O	O	O	O	O	O
Nature of address	-	-	X	O	O	O	O	O	O	O	-	-	-	-	-	-	-
Bearer capability	X	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O

Key:
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 FacilityInformationElement contained in a signalling message as defined in DSS1 and ISUP.
NOTE 3: In a local exchange only. The BCSM has to analyze (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 DP3-10 in the OBCSM for a party served by an ISDN BRI or PRI and can be available at DP8 in the OBCSM for a party served by a non-ISDN line. A feature activation/indication can be available at DP14-18 in the TBCSM for a party served by an ISDN BRI or PRI and can be available at DP16 in the TBCSM 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.

If a criteria is marked with a "X" for a DP, then this means that a conditional TDP which is armed at the DP may require the criteria as listed in the table to be satisfied before informing the SCF that the TDP was encountered. E.g. a conditional TDP at DP1 may require the class of service criteria to be satisfied before the SCF is informed that the TDP was encountered.

If a criteria is marked with an "O" for a DP, 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 a criteria marked with a "X".

Concerning impact on memory and real time performance, see Clause C.1.

DP of the OBCSM and TBCSM

DP1	OriginatingAttemptAuthorized
DP2	CollectedInfo
DP3	AnalyzedInformation
DP4	RouteSelectFailure
DP5	OCalledPartyBusy
DP6	ONoAnswer
DP7	OAnswer
DP8	OMidCall
DP9	ODisconnect
DP10	OAbandon
DP11	Reserved
DP12	TerminatingAttemptAuthorized
DP13	TCalledPartyBusy
DP14	TNoAnswer
DP15	TAnswer
DP16	TMidCall
DP17	TDisconnect
DP18	TAbandon

C.2.3 Trigger criteria combinations

A particular conditional TDP may require a combination of the applicable criteria to be checked before informing the SCF that the TDP was encountered.

Criteria may be combined using the:

- AND function (e.g. at DP3, initiate an SCF enquiry if "Class of Service" = international call barred AND "Nature of Address" = international);
- OR function (e.g. at DP3, initiate an SCF enquiry if "Called Party Number" = 831 or "Called Party Number" = 836);
- NOT function (e.g. at DP3, initiate an SCF enquiry if "Called Party Number" is NOT = 831).

The criteria combinations to be checked for a particular TDP is to be specified by means of management procedures applicable for the CCF/SSF.

C.2.4 Trigger criteria priorities

Multiple TDP-Rs may be armed at the same DP, each TDP-R being either an unconditional or a conditional TDP. Each TDP-R will have an associated set of criteria (i.e. conditions specific to the particular TDP-R that shall be met before the SCF is informed that the TDP is encountered). The priority of the TDP-Rs at a DP is to be specified by means of management procedures applicable for the CCF/SSF.

During the processing of TDP-Rs at a DP, the SSF has to check the set of criteria associated with each TDP-R in descending order of priority until either the set of criteria for a particular TDP-R are met (resulting in informing the SCF) or all sets of criteria have been checked and none have been met.

Annex D (informative): Bibliography

1) ITU-T Recommendation Q.711 (1993): "Functional description of the signalling connection control part".

2) ITU-T Recommendation Q.763 (1993): "Formats and codes".

3) CCITT Recommendation Q.767 (1991): "Application of the ISDN user part of CCITT signalling system No.7 for international ISDN interconnections".

4) ITU-T Recommendation Q.771 (1993): "Specifications of Signalling System No.7; Functional description of transaction capabilities".

5) ITU-T Recommendation Q.774 (1993): "Specifications of Signalling System No.7; Transaction capabilities procedures".

6) ITU-T Recommendation Q.932 (1993): "Digital subscriber Signalling System No.1 (DSS1) - Generic procedures for the control of ISDN supplemenatary services".

7) ETR 090 (1993): "ETSI object identifier tree; Common domain; Intelligent Network (IN) domain".

8) ETS 300 374-5: "Intelligent Network (IN); Intelligent Network Capability Set 1 (CS1); Core Intelligent Network Application Protocol (INAP); Part 5: Protocol specification at the SCF-SDF interface".

History

Document history	
September 1994	First Edition
February 1996	Converted into Adobe Acrobat Portable Document Format (PDF)