ETSI TECHNICAL COMMITTEE REFERENCE TECHNICAL REPORT

TCR-TR 001

March 1992

Source: ETSI TC-NA

Reference: DTR/NA-60106

ICS:

Key words: IN, Network, Framework

Network Aspects (NA);
Intelligent Network: Framework

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

1 Objectives and scope of the IN framework document

The purpose of this TCR-TR is to provide a stable basis for the development of IN standards.

It contains information regarding IN Concepts and Modeling:

- IN objectives, definition and scope
- IN overall requirements
- IN conceptual model
- Service Plane Architecture
- Global Functional Plane Architecture
- IN functional architecture (high level)
- IN Call modeling (high level)

This TCR-TR records the current status of study into IN within ETSI NA6 up to September 1991 and identifies the aspects which could be standardized.

IN will be standardized in evolution phases. The set of technical capabilities to be standardized in a phase called Capability Set (CS). The technical capabilities to be standardized for CS1 are identified in TCR/TR2 "Guidelines for Standards in IN CS1".

TCR-TR1 content is mapped to the content of the Q.12xy series of the CCITT Recommendations as indicated in the table below:

TCR-TR1 Chapter	CCITT Rec.	Title
2 to 4.2	Q.1201	Principles of IN Architecture-Chapter 1,2 and 3
4.3	Q.1202	IN Service Plane Architecture
4.4	Q.1203	IN Global Functional Plane Architecture
4.5	Q.1204	IN Distributed Plane Architecture
4.6	Q.1205	IN Physical Plane Architecture
5	Q.1201	Principles of IN Architecture-Chapter 4 (IN Long Term Architecture Framework)
6	Q.1290	Vocabulary

2 Objectives, overall description

2.1 Motivation, objectives, scope of Intelligent Network

2.1.1 Motivation

The term Intelligent Network (IN) is used to describe an architectural concept which is intended to be applicable to all telecommunications networks. IN aims to ease the introduction of new services (i.e. Universal Personel Telecommunication (UPT), Virtual Private Network (VPN), Freephone ....) based on more flexibility and new capabilities.
IN recommendations are motivated by the interests of telecommunication service providers to rapidly, cost effectively and differentially satisfy their existing and potential market needs for services. Also, these service providers seek to improve the quality and reduce the cost of network service operations and management.

Additionally, current trends in technology permit a greater degree of intelligence and greater freedom in the allocation of intelligence in the telecommunications network. For example, the improved mobility derived from miniaturization of electronic components allows for a greater degree of distributed functionality within and between service provider networks. Factors permitting such intelligence include advances in digital transmission and switching, common channel signalling, distributed data processing, database management and expert systems.

2.1.2 Objectives of Intelligent Network

The objectives of IN is to allow the inclusion of additional capabilities to facilitate service/network implementation independent provisoning of services in a multi-vendor environment. Service implementation independence allows service providers to define their own services independent of service specific developments by equipment vendors.

Network implementation independence allows network operators to allocate functionality and resource within their networks and implementation specific develoment by equipment vendors.

2.1.3 Scope of Intelligent Network

Types of networks: IN is applicable to a wide variety of networks, including (but not limited to): public switched telephone network (PSTN), mobile, packet switched public data network (PSPDN) and Integrated Services Digital Network (ISDN) - both Narrowband-ISDN (N-ISDN) and Broadband-ISDN (B-ISDN).

Type of services: IN supports a wide variety of services, including supplementary services, and utilizes existing and future bearer services (e.g., as those defined in N-ISDN and B-ISDN contexts).

The types of networks and services that will be supported will be identified in a subsequent TR.

A number of factors that may influence the scope of the work on IN were identified as:

- Activities on ONP
- Network Operator plans
- Industrial Product offerings
- Market needs

2.2 Definition of Intelligent Network

Intelligent Network (IN) is an architectural concept for the operation and provision of new services which is characterized by:

- extensive use of information processing techniques;
- efficient use of network resources;
- modularization and reusability of network functions;
- integrated service creation and implementation by means of the modularized reusable network functions;
- flexible allocation of network functions to physical entities;
- portability of network functions among physical entities;
- standardized communication between network functions via service independent interfaces;
- service subscriber[1] control of some subscriber-specific service attributes;
- service user control[2] of some user-specific service attributes;
- standardized management of service logic;

2.3 Evolution of Intelligent Network Recommendations

A phased standardization process is recommended. This recommendation takes into account the fact that the specification and the deployment of networks that meet all the objectives of the IN target architecture will take many years. In addition, IN as a new architectural concept should be introduced starting from the existing networks and the current recommendations.

Furthermore, the IN target (long term views) will evovle, reflecting operational experiences, new technological opportunities and market evolution.

In order to ensure smooth evolution towards the target, IN recommendations shall allow

- backward compatibility of each evolutionary phase,
- open-endedness towards long term views.

In particular, backward compatibility implies utilization of previous recommendations within a new phase.

---

1) A service subscriber (customer) is a person or entity who, or which, obtains a service from a service provider and is responsible for the payment of the charges due to that service provider.

2) A service user is a person who has access to and makes use of services.

2.3.1 General considerations on the standardization process

Figure 1 shows how the different aspects of the standardization process are related, distinguishing

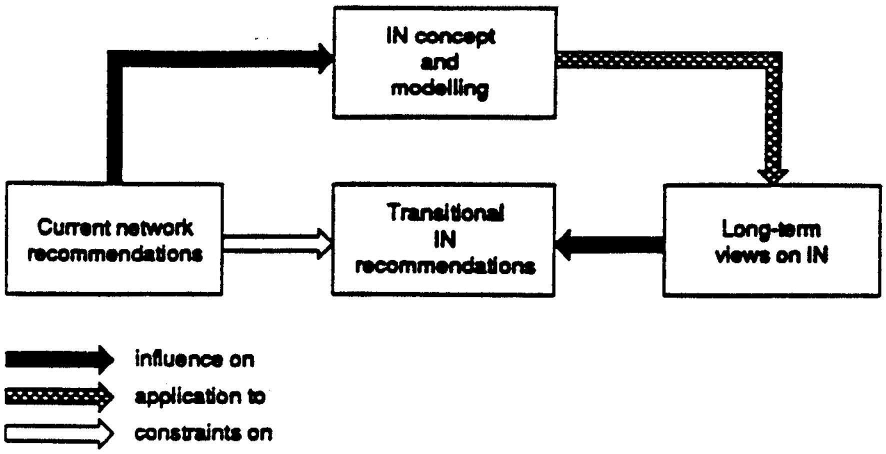


Figure 1: Relationship between the aspects of the standardization process

- IN concept and modelling, which are influenced by the current network recommendations; and

- their application to long-term views and to recommendations on intermediate phases (transitional IN recommendations).

The influence of current network recommendations and of long term views on transitional IN recommendations is also shown.

2.3.2 Recommendations areas

A recommendations outline can be derived from the above described standardization process. Three areas are fundamental to the production of recommendations:

Area 1: IN architectural concept and modelling
This area contains the IN concept, modelling techniques and other network design tools, as well as the results from the develoment of the IN target architecture.

Area 2: IN transition planning and phase definitions
Here guidelines are worked out allowing a proper transition from the existing technology base towards a target IN infrastructure. For each phase it is necessary to determine the service functionality (e.g., whether or not to include service creation capabilities for the customer) and technological constraints (e.g.,using the D-channel for ISDN packet switched services).

Area 3: <u>IN architecture and interfaces for each phase</u>
In this area the specifications are provided that are necessary for the implementation of IN equipment, interfaces, etc. For each phase an evolving set of recommendations will be develoed.

The relationship between the areas is shown in Figure 2.

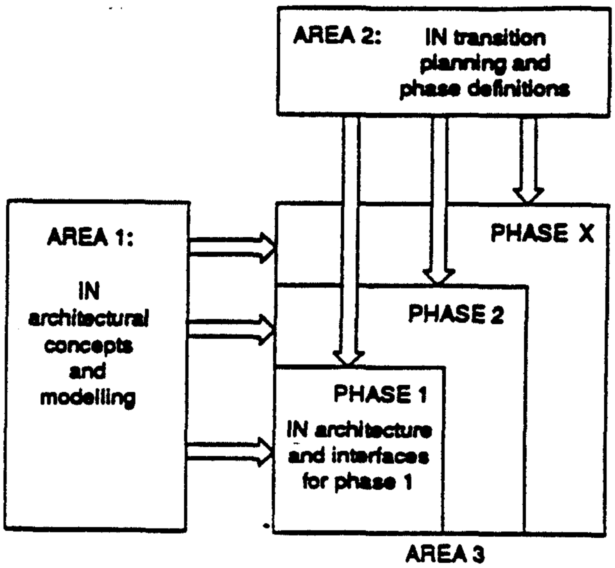


Figure 2: Recommendations areas

2.3.3 Phased standardization and definition of capability sets

Capability Sets (CS) are defined as sets of IN capabilities which are to be subjects of standardization activities and for which the availability of recommendations will be targeted for a particular evolution phase.

The "Long Term Capability Set" (LTCS) is the CS the Target IN Architecture.

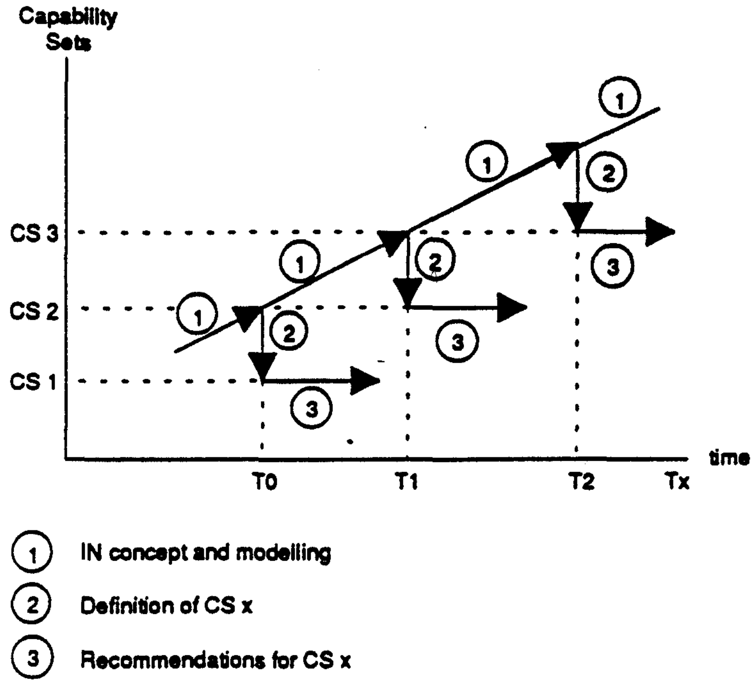


Figure 3: Sequencing of capability sets (CSs)

The sequencing of CSs is shown in Figure 3. The figure also indicates the relationship between the previous defined areas and the definition of each Capability Set.

2.3.4 Working Procedure

The development of IN ETSs can be split into 3 different, consecutive tasks:

Task 1) defining an architecture and a comprehensive set of capabilities for the target, long term IN, as far as current knowledge allows;

Task 2) defining a subset of capabilities, to be further detailed, for each phase of IN, consistent with the architecture defined in Task 1;

Task 3) detailing the capabilities identified in Task 2, in order to produce implementable standards.

The consequence of the proposed way of IN-standardisation is:

1) Although phases cannot be defined beforehand, an implicit phased strategy is still used (phases can be concluded afterwards).

2) Backwards compatibility can be quaranteed.

The following IN standards/stages are in principle, super sets of their predecessors.

Note:

- The TCR-TR "IN Framework" relates to Area 1.
- The TCR-TR "Guidelines for Standards on IN CS1" relates to Area 2.
- The means for identifying the appropriate documentation for the specific phases of Area 3 is FFS.

3 IN functional requirements

3.1 Introduction

IN functional requirements arise as a result of the need to provide network capabilities for both:

- customer needs (service requirements); and
- network operator needs (network requirements).

A service user is an entity external to the network that uses its services. A service is that which is offered by an administration to its customers in order to satisfy a telecommunications requirement. Part of the service used by customers may be provided/managed by other customers of the network.

Service requirements will assist in identifying specific services that are offered to the customer. These service capabilities are also referred to as (telecommunication) services. Network requirements span the ability to create, deploy, operate and maintain network capabilities to provide services. The categorization of service requirements versus network requirements is schematically shown in Figure 4.

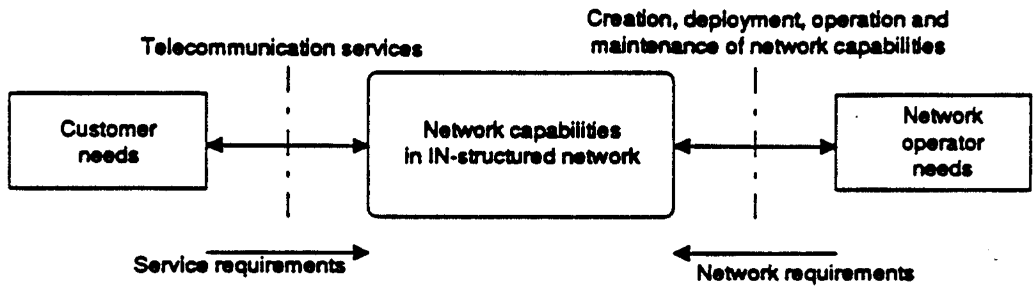


Figure 4: Service requirements vs. network requirements

Service and network requirements can be identified for the following areas of service/network capabilities: service creation, service management, network management, service processing and network interworking.

- Service creation: is an activity whereby supplementary services are brought into being through specification phase, development phase and verification phase.
- Service management: is an activity to support the proper operation of a service and the administration of information relating to the user/customer and/or the network operator. Service management can support the following processes: service development, service provisioning, service control, billing and service monitoring.
- Network management: is an activity to support the proper operation of an IN-structured network.
- Service processing: consists of basic call and supplementary service processing which are the serial and/or parallel executions of network functions in a coordinated way, such that basic and supplementary services are provided to the customers.

- Network interworking: is a process through which several networks (IN to IN or IN to non-IN) cooperate to provide a service.

Figure 5 gives a general overview of these capability areas including their relation to service and network requirements. The network interworking capabilities are not shown in this figure as these are indirectly contained in the other capability areas.

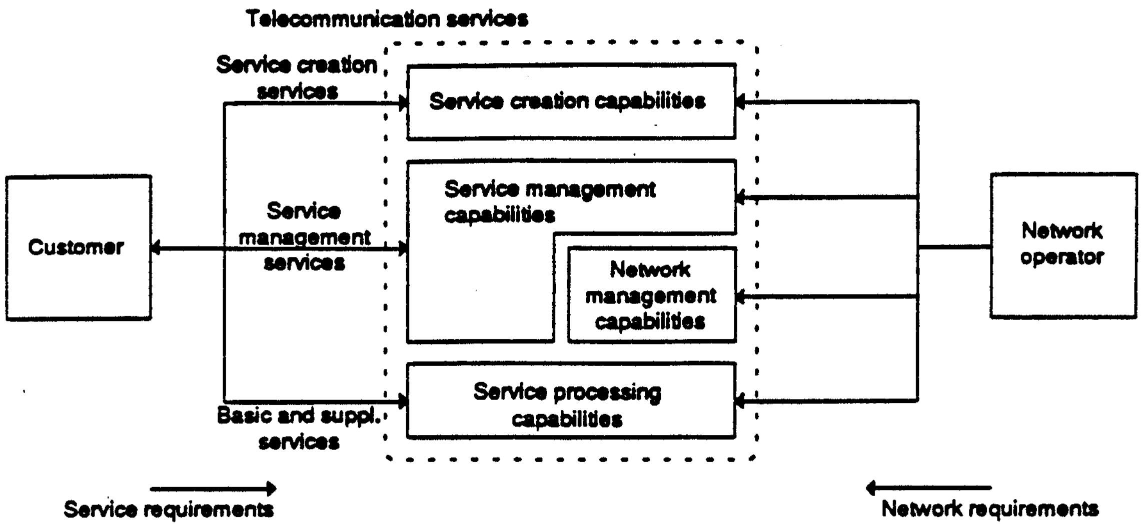


Figure 5: Network capabilities in IN-structured network

3.2 Service requirements

3.2.1 Overall requirements

The following requirements may also apply to existing networks. Nevertheless, they are stated here to underline their importance when defining the IN architecture.

- it should be possible to access services by the usual user network interface (e.g., POTS, ISDN);
- it should be possible to access services that span multiple networks;
- it should be possible to invoke a service on a call by call basis, or for a period of time, in the latter case, the service may be deactivated at the end of the period;
- it should be possible to perform some access control to a service;
- it should be easy to define and introduce services;
- it should be possible to support IN-supported supplementary services involving calls between two or more parties;
- it should be possible to record service usage in the network (service supervision, tests, performance information, charging);
- it should be possible to provide services that imply the use of functions in several networks;

- it should be possible to control the interactions between different invocations of the same service.

3.2.2 Service creation

A subset of the service creation capabilities used by the network operator (described in section 2.2.2) may be offered to customers. Service creation refer to the network capabilities that are used by network operators for the provision of services to customers. This is schematically shown in Figure 6.

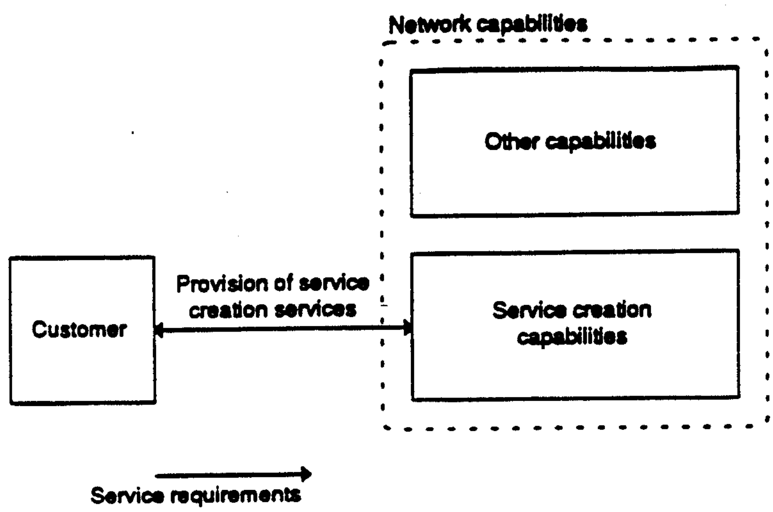


Figure 6: Service requirements for service creation

3.2.3 Service management

Service requirements for service management refer to the network capabilities that are necessary for the provision, from a customer point of view, of service management services to customers. This is schematically shown in Figure 7.

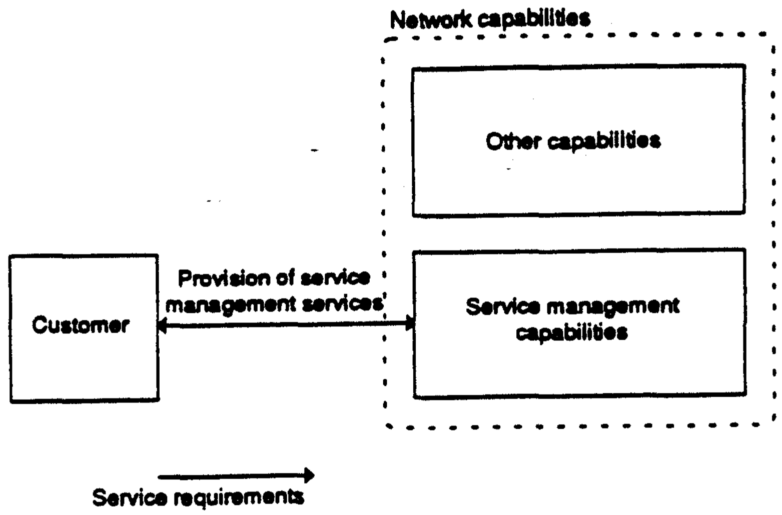


Figure 7: Service requirements for service management

A subset of the service management capabilities used by the network operator may be offered to customers.

3.2.3.1 Service management during deployment phase

(For further study.)

3.2.3.2 Service management during provisioning phase

Service provisioning is the activity of installing and deploying the necessary functionality in appropriate network elements to realize a service to a specific customer along with the initial activation and customization. After provisioning the customer's service is administered.

3.2.3.3 Service management during utilization phase

This activity includes:

- activation, deactivation, service maintenance and service customization, after the service provisioning has taken place.

- service activation which is the activity to make the service usable by a specific customer (e.g. call forwarding activation).

- service customization which is the activity of setting up the appropriate service parameters to control the operation of service to meet the specific needs of the customer (e.g. setting Call Distribution percentages).

- charging which is mainly to collect data on service usage and to generate reports there on for billing either on demand or automatically. It includes the alteration of charge within the framework of agreement with a network operator. Other requirements include the preparation of customer-specific billing reports and data which are accessible to the customer.

- service monitoring provides the capability to collect and accumulate statistics on a given service with a view to determine the Quality of Service operation and adjust the operation to suit the prevailing conditions; also the data may be used in the process of service creation to determine if an implementation of an IN supported supplementary service meets the service's performance requirements.

3.2.4 Service processing

Service requirements for service processing refer to the network capabilities that are necessary for the provision, from a customer point of view, of basic and supplementary services by an IN-structured network. This is schematically shown in Figure 8.

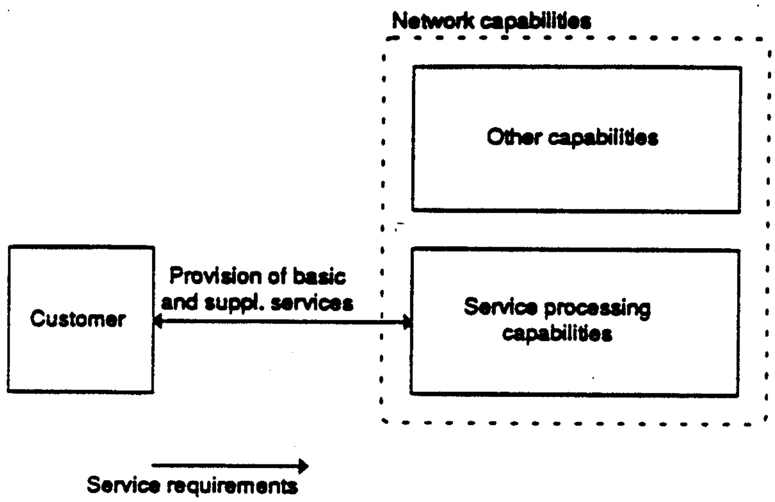


Figure 8: Service requirements for service processing

The IN is primarily a network concept that aims for efficient creation, deployment and management of supplementary services that enhance basic services. With regard to the provision of basic and supplementary services itself the IN concept is "transparent" to the customer, i.e. the customer is unaware of whether a service is provided in an IN way or not. This "transparency" basically implies that, from a customer point of view, no service processing requirements can be identified that have specific reference to the IN as such. Not withstanding this, the IN should be capable of supporting a broad range of basic and supplementary services. Service processing requirements can be identified for the following capabilities:

- service capabilities that are necessary to support a broad range of basic and supplementary services;

- access capabilities that are necessary to interface with the network and to access the services.

The relationship between service capabilities and access capabilities is visualized in Figure 9.

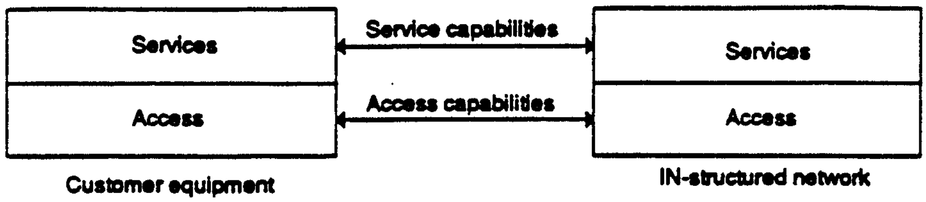


Figure 9: Service capabilities and access capabilities

3.2.4.1 Service capabilities

Requirements on service capabilities directly result from the services that are to be supported by the network. Service capabilities are related to basic services (bearer and tele-) and, in particular, to the supplementary services that may enhance these basic services. As stated above, the IN is primarily a network concept for the support (creation, deployment, etc.) of supplementary services. Because supplementary services can only be provided in combination with basic services, it is necessary to define for which basic services the IN concept can be applied. In this respect the IN concept can be applied to the support of supplementary services for the following basic services:

Bearer services:

- circuit-mode unrestricted (various bit rates)
- circuit-mode speech
- circuit-mode audio
- packet switched data services
- circuit switched data services
- others

Teleservices:

- telephony
- telefax
- Videotex

Broadband interactive services:

- Conversational service

- Messaging services
- Retrieval services

Broadband distribution services:

- Distribution services without user individual presentation control
- Distribution services with user individual presentation control
- others

With regard to the provision of supplementary services the network should support service-independent network functions in such a way that will facilitate rapid service introduction.

3.2.4.2 Access capabilities

In order to use a particular service, the customer needs an access arrangement to the network(s), that is providing the service. Nevertheless, the customer is mainly interested in the services themselves and not in the specific access arrangement that is used to physically connect to the network. For example, it should be possible that a single Virtual Private Network service can be accessed by various access arrangements such as ISDN interfaces, POTS interfaces, "mobile" interfaces, etc. In this case the VPN service could span multiple networks with different interface technologies. Access arrangements via (non-IN) sub-networks should also be supported (see Figure 10). In case of multiple access arrangements to a service, it should be noted that a particular access arrangement may impose technical, operational and/or regulatory limitations on the provided service.

When a particular service spans multiple networks, network interworking arrangements will be required between the different networks to allow ubiquitous provision of the service. Requirements for network interworking capabilities are identified in section 2.2.6.

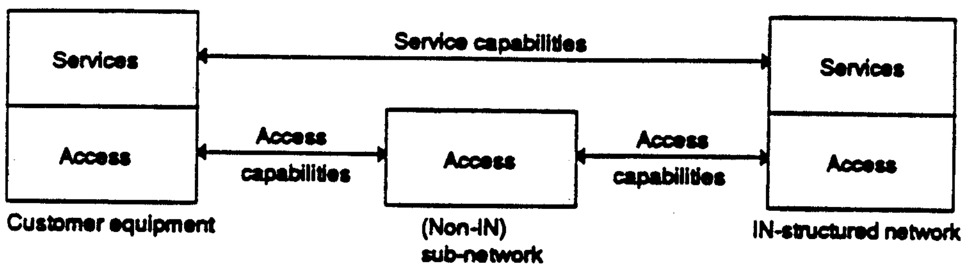


Figure 10: Access to services via (non-IN) sub-networks

In order to allow freedom of access to services, it is necessary that there is a sufficient degree of independence between the service capabilities on the one hand and the access capabilities on the other hand. The following access capabilities are foreseen for the IN:

fixed network:

- PSTN access
- ISDN access

- PSPDN access
- private network access

mobile network:

- PLMN access

broadband network

- Asynchronous Transfer Mode (ATM) access
- Synchronous Transfer Mode (STM) access

3.2.5 Service interworking

Service interworking describes special aspects of the individual service, if the service is used in a connection which exists partly inside a given IN-structured network and partly inside a non-IN structured network, or which, for certain operational aspects, routes through more than one IN structured network.

3.3 Network requirements

3.3.1 Overall requirements

The following reflects overall network requirements. Specific detailed requirements can be found in the relevant sub-section:

- it should be possible to move cost-effectively from existing network bases to target network bases in a practical and flexible manner;
- it should be possible to reduce redundancies among network functions in physical entities;
- it should be possible to allow for the flexible allocation of network functions to physical entities;
- there is a need for communication protocols that allow flexibility in the allocation of functions;
- it should be possible to create new services from network functions in a cost and time efficient manner;
- it should be possible to guarantee the integrity of the network when a new service is being introduced;
- it should be possible to manage network elements and network resources such that Quality of Service and network performance can be guaranteed.

3.3.2 Service creation

Network requirements for service creation refer to the network capabilities that are necessary, from a network operator point of view, for the creation of new supplementary services. This is schematically shown in Figure 11.

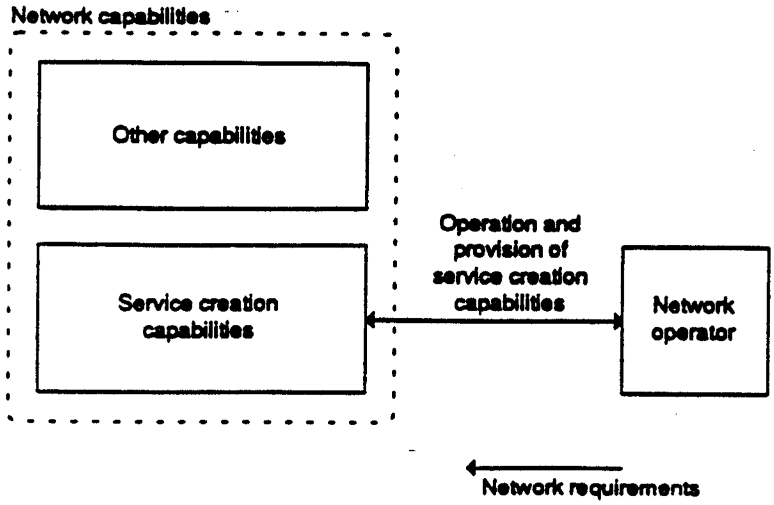


Figure 11: Network requirements for service creation

The creation of (new) services consists of several steps which are summarized in the service creation process. The different steps are:

- service specification;
- service development;
- service verification.

Service specification is the first step in the service creation process. As such, this step includes such activities as refinement of detailed service description requirements, functional analysis, generation and verification of a service specification and definition of a high level structured design. The primary outputs of this step include a service specification and a high level structured design which together provide sufficient detail to drive subsequent service development and verification steps.

Service development is the step which transforms a high level structured design into a detailed structured software design and subsequently develops the necessary software components, data definitions, etc. required to realize that design. The major output of this step is the developed service software and documentation which is ready for more rigorous service verification testing.

Service verification is the step in the service creation process where the developed service software (including supporting documentation) is rigorously tested to validate that the resulting service application completely satisfies the specification. The principal output of this step is thus the verified service software and supporting documentations required for deployment.

The Service Creation Environment provides an environment which allows the easy creation of (new) services in a network-configuration and network-type independent way by means of service independent building blocks.

The Service Creation Environment should, in an efficient and effective way, provide service-implementation independent tools, techniques, languages (e.g. specification languages) and procedures to support the service creation process in which service scripts, represented for example by Service Logic Programmes, Service Management Programmes and Data Template Programmes, can be created.

There is a need to standardize a representation of service logic with standardized function calls (Application Programming Interface) to IN capabilities.

In order to guarantee the integrity and security of the networks as well as to guarantee the integrity of each created service, it should be necessary to define the scope of accessible IN capabilities through the service creation activity.

This scope isolates the unnecessary and unsolicited interactions between service and accesses to the network capability.

Each service can be represented by several different types of service scripts. Each script may be classified according to certain characteristics, based on its type of usage. These scripts must all be implemented on a service-, network configuration-, and network-type (e.g. PSTN, ISDN, PLMN) independent way. This can best be done by means of service independent building blocks. These building blocks are combined by means of sequential and conditional programming statements using selected programming languages.

Since different scripts can be identified also different building blocks (related to the scripts) can be identified.

Examples of service scripts are:

- Service Logic Program (SLP)

An SLP consists of, for example, service execution logic, statistics logic, data base logic, charging logic, etc. SLP is a "real time" program. E.g., an SLP logs information concerning billing and statistics into a data base.

- Service Management Program (SMP)

An SMP consists of for example statistics logic, charging logic, data base logic, etc. SMP is a "batch oriented" program which, for example, creates enhanced charging and service statistic reports derived from the logs created by an SLP. SMP supports the Service Management and not the Network Element or Network Management.

- Data Template Program (DTP)

A DTP consists of service-data related logic, user-data related logic, data base related logic, etc. DTP is an "interactive" program, which interacts with the network operator (creating and using service-data templates) or the customer (creating and using user-data templates) and updates the service and user data.

Network operators may choose to offer a subset of the service creation capabilities to service users. If this is the case the following requirements apply:

- It should be able to define service user classes (e.g. some user can create and use network wide services, some users can create and use local services closed in their own local area, some users can create and use ATM related services, and some users cannot, etc.).

- In order to define service user classes, the restriction is defined with a particular set of network resources (restricted-network resources) for particular users. Network resources have different levels of abstraction.

3.3.3 Service management

Network requirements for service management refer to the network capabilities that are necessary, from a network operator point of view, to support the proper operation of services. This is schematically shown in Figure 12.

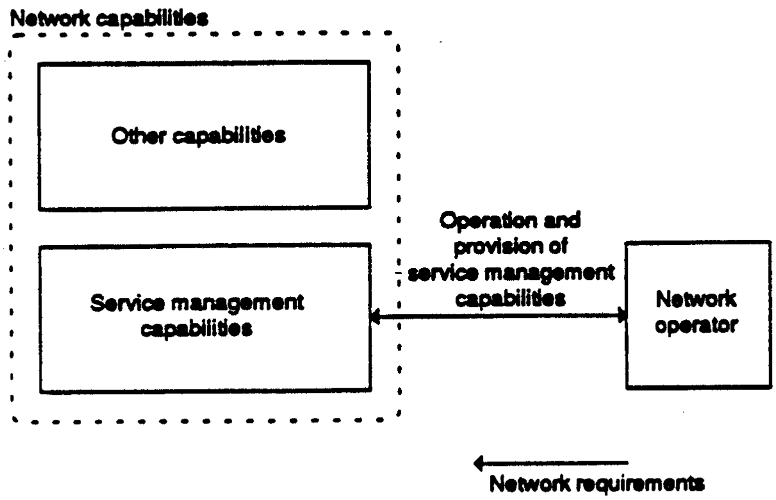


Figure 12: Network requirements for service management

3.3.3.1 Service management during deployment

This activity requires the following:

- limited deployment of service for testing purposes. If testing is successful follow with
- downloading of service logic, management logic, etc., into appropriate network elements, and
- activation of logic.

3.3.3.2 Service management during provisioning phase

This involves such activities as the creation of the customer's service profile in the service logic and the activation of the service for the customer. This may require the initialization of trigger conditions and the creation of the customer's service profile in the service logic.

3.3.3.3 Service management during utilization phase

- Service control gives the customers the ability to modify parameters which control their service. This could include, for example, screening lists or rules for routing calls by time of day and/or day of week.
- Data entered by the customer must be validated to insure that, for example, the customer has the permission to change the indicated parameters and the parameters, as modified, are valid.

- Under IN, there may need to be several kinds of charging schemes,e.g., flat rate and usage sensitive. The usage sensitive charging may consider not only the duration of a call, but also the network resources required to provide the service. In this latter case, the tariff may be "service independent" in the sense that is computed from the cost of the network resources used rather than from the value to the customer of the service offered.

Service monitoring can be either automatic or manual. Typical manual operations include queries regarding status information and network configuration.

Data gathered automatically could include traffic loads and other performance data. The following activities support automatic service monitoring:

- create and modify measurement schedules which include such fields as the measurements to be made, the frequency of data collection for each measurement, and the format and frequency of each report.. ;

- collect and validate data requested by the schedules;

- format and deliver the reports as requested;

3.3.4 Network management

Network requirements for network management refer to the network capabilities that are necessary, from a network operator point of view, to support the proper operation of the IN-structured network. This is schematically shown in Figure 13.

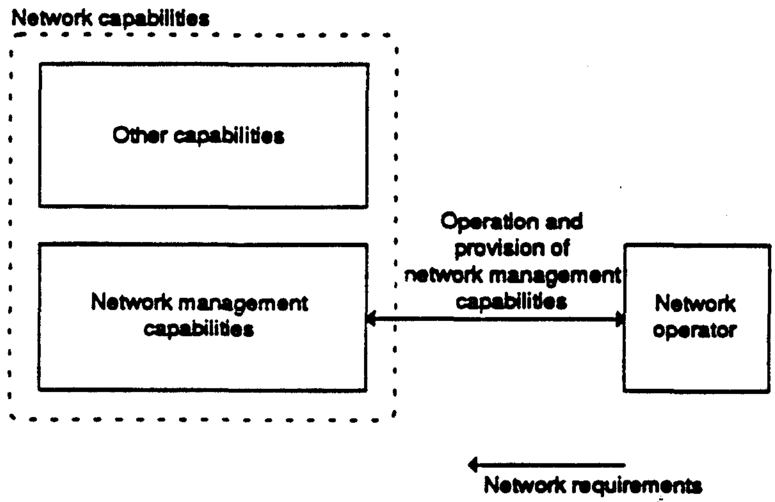


Figure 13: Network requirements for network management

The IN and non-IN requirements of network management are not essentially very different. The TMN application functions are relevant to IN as much as they are to non-IN. So, the following organization of management capabilities (Recommendation M.30) are applicable.

1) Performance management
   - performance monitoring
   - traffic management and network management
   - Quality of Service observations

2) Fault (maintenance) management
   - alarm surveillance
   - failure localization
   - testing

3) Configuration management
   - provisioning
   - status and control
   - installation

4) Accounting management

5) Security management

Additional application functions may be needed to deal with the IN situation.

3.3.5 Service processing

Network requirements for service processing refer to the network capabilities that are necessary for the provision, from a network operator point of view, of basic and supplementary services by an IN-structured network. This is schematicallv shown in Figure 14.

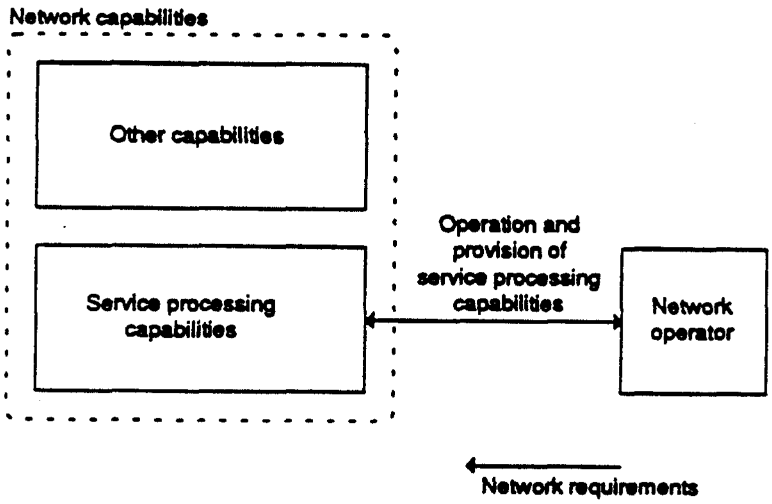


Figure 14: Network requirements for service processing

The main network requirements for service processing stem from the inability of network operators of traditional "non-IN" networks to rapidly create and deploy new supplementary services. To overcome this inability the IN aims for:

- rapid service implementations by means of reusable network functions;
- modularization of network functions;
- standardized communication between network functions via service independent interfaces.

To better understand how the IN can achieve the goal of fast service implementation, an overall comparison is made between traditional non-IN service processing (not meeting the aims listed above) and newly required IN service processing (meeting the aims listed above). Both cases are respectively referred to as the non-IN service processing model and the IN service processing model.

Non-IN service processing model

A simplified but typical representation of a non-IN service processing model is illustrated in Figure 15.

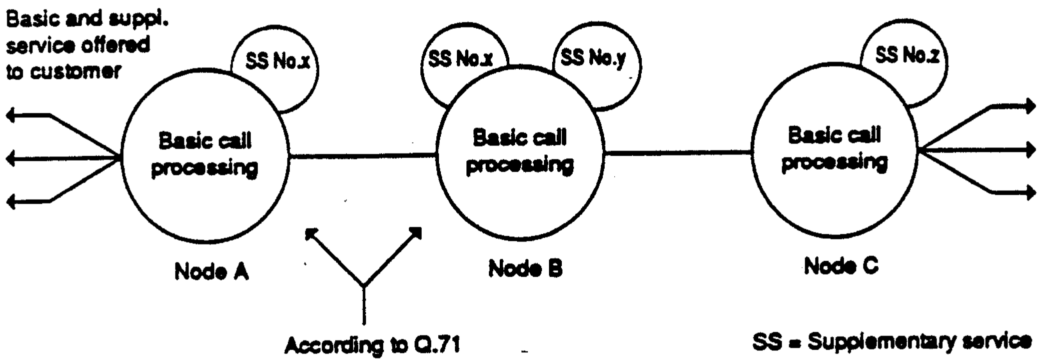


Figure 15: Non-IN service processing model

The main elements in this model are the basic call processes (in each node) and the extensions on these processes for supplementary services. The model is consistent with Q.65 ("Stage 2 of the method for the characterization of services supported by an ISDN") and Q.71 ("ISDN 64 kbit/s circuit mode switched bearer service"). Some important characteristics of the non-IN service processing model are the following:

- the basic call process supports the provision of a particular basic service and is designed for optimal Quality of Service and network performance;
- each node contains an instance of the basic call process;
- in each node the basic call process can be modelled as a Finite State Machine that communicates with "neighbour" basic call processes;
- each supplementary service extension is a monolithic non-reusable entity that modifies or supplements the basic call process;
- there is no reusability of service-independent network functions;
- there is no fast service implementation.

For each introduction of a new supplementary service the basic call process needs to be modified/supplemented. Generally this is a very time-consuming process because each new service requires new software generics to be loaded on the individual nodes.

As a conclusion the non-IN service processing model does not allow fast service implementation.

IN service processing model

A high level overview of a desirable IN service processing model is illustrated in Figure 16.

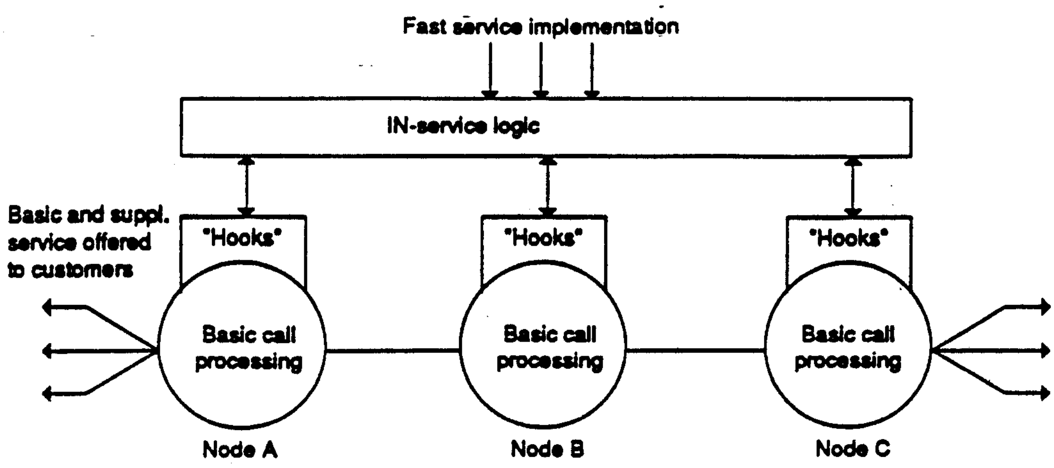


Figure 16: IN service processing model

The main three elements of this model are: the basic call processes, the "hooks" that allow the basic call processes to interact with IN service logic, and IN service logic that can be "programmed" to implement new supplementary services. For these elements the main principles are described below:

- The basic call process should be available all over the network and is designed to support, with optimal performance, services that do not require special features. In order to achieve flexibility in service processing, the basic call process needs to be modularized into service-independent sub-processes such that these can be executed autonomously (without interference from the outside during execution).

- "Hooks" are to be added to the basic call process forming the links between the individual basic call sub-processes and the service logic. The "hooks" are able to start an interaction session with the IN service logic. For this it should continuously check the basic call process for the occurrence of conditions on which an interaction session with IN service logic should be started. During an interaction session the basic call process can be temporarily suspended.

- IN service logic uses a programmable software environment that needs to be developed to allow fast implementation of new supplementary services. New supplementary services can be created by means of "programs" containing IN service logic. The IN service logic is able, via the "hooks" functionality, to interact with the basic call process. In this way IN service logic can control the sub-processes in the basic call process and the sequencing of these sub-processes.

- Thus, by changing logic at the service control point and modifying network data, a new service that uses existing network capabilities can readily be implemented.

- In addition IN service logic can decide to terminate an interaction session with the basic call process. The basic call process will then resume its execution as specified by the IN service logic. In order to allow fast service implementation, the IN service logic should have a logical view of the network resources that constitute the basic call process and additional (specialized) network functions.

- For proper service processing, the following principles apply:
  - it should be possible to distribute resources between services in a well balanced way;
  - it should be possible for IN supported services to share resources with non-IN supported services;
  - it should be possible to provide the different way of resource data management from current embedded way;
  - it should be possible to introduce IN supported services specific resources.

Consequences for call modelling in an IN architecture

As part of the activities to define an IN architecture including the network elements within this architecture there is a need for a call model that describes the real time behaviour of call control capabilities for the provision of basic and supplementary services. In order to be consistent with the principles of the above-described IN service processing model, the IN call model should cover the following aspects:

- it should specify which basic services can be supported by the model;
- it should model the basic call processes (each individual basic service may require its own IN basic call process);
- it should describe trigger mechanisms ("hooks") that allow the IN basic call process to interact with service logic;
- it should provide a logical view (from service logic point of view) of call. processing functions and network resources, in this way allowing fast service implementation;
- it should specify the mechanisms according to which an IN basic call process may interact with the service logic (e.g. single-ended interactions, simultaneous interactions, service-logic initiated interactions, etc.);
- it should be evolvable from the existing technology base.

3.3.6 Network interworking

Network interworking is a process in which several networks (IN to IN or IN to non-IN) cooperate to provide a service.

The need for network interworking capabilities results from the fact that customers may want to access services which span multiple networks. These networks may have different access types (PSTN, ISDN, etc.) and may have different levels of IN structuring (full, partial or no IN structuring). Irrespective of the access type and of the level of IN structuring services should be provided to customers in a consistent way.

Network interworking requirements exist at different levels:

- service processing
- service management
- service creation

and in each level, some network interworking related gateway functions need to be defined.

3.3.6.1 Gateway functions for service processing

Figure 17[1][2] represents possible "gateway" functions which support network interworking at the level of service processing, when two IN-structured networks cooperate to provide a service.

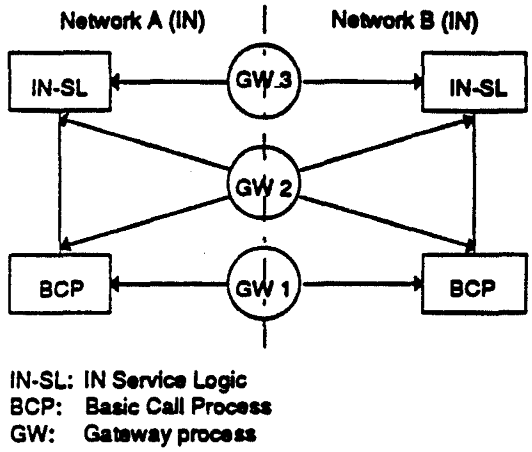


Figure 17: Possible gateway functions between two INs

A gateway function may be used, for example, to access service logic in other networks (GW2), or to provide communication between pieces of service logic pertaining to different networks (GW3). GW1 is used to route the call between the networks.

1) It is anticipated that substantial service capabilities will result from earlier implementation of GW1 and GW3.

2) GW2 requires careful further study in particular from network security and integrity point of view.

Figure 18 represents the use of network interworking between an IN-structured and a non-IN structured network.

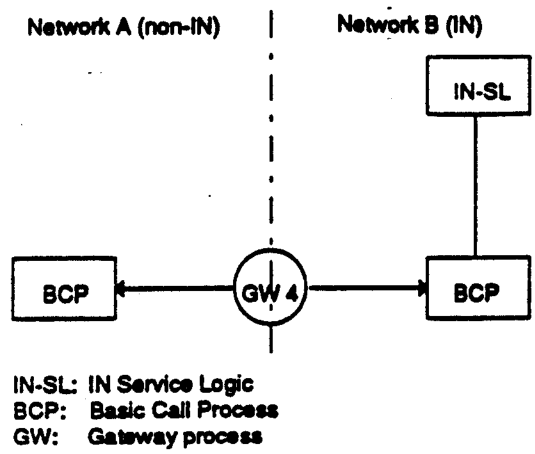


Figure 18: Gateway function between IN and non-IN

A gateway function (GW4) is required to route the call between the IN-structured and non-IN-structured networks and to provide interworking between the BCP of network A and the BCP of network B.

3.3.6.2 Gateway functions for service management

At this level, a gateway function is required to link the service management process of the different networks which interwork. The gateway function should support service management during deployment, provisioning and utilization phases for services that span multiple networks.

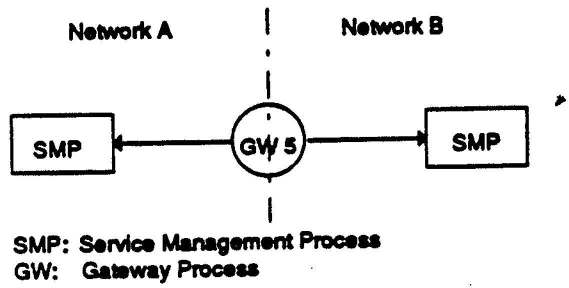


Figure 19: Gateway function for service management

3.3.6.3 Gateway functions for service creation

Service creation is an off-line activity and normally exists in a single network. The need for network interworking at the service creation level is for further study.

4 IN architectural concept

4.1 Requirements and Assumptions

A key objective of the IN is to provide service-independent functions that can be used as "building blocks" to construct a variety of services. This allows easy specification and design of new services.

A second key objective is network implementation independent provision of services. This objective aims to isolate the services from the way the service-independent functions are actually implemented in various physical networks, thus providing services that are independent of underlying physical network infrastructure(s).

The network implementation independence has these objectives:

- services use distributed network functions in various ways;
- services can span several networks and are independent of specific implementations in these networks;
- services are independent of technological developments and evolution in network infrastructure, so that physical networks can evolve without affecting existing services;
- physical elements in such a network can be procured from different vendors.

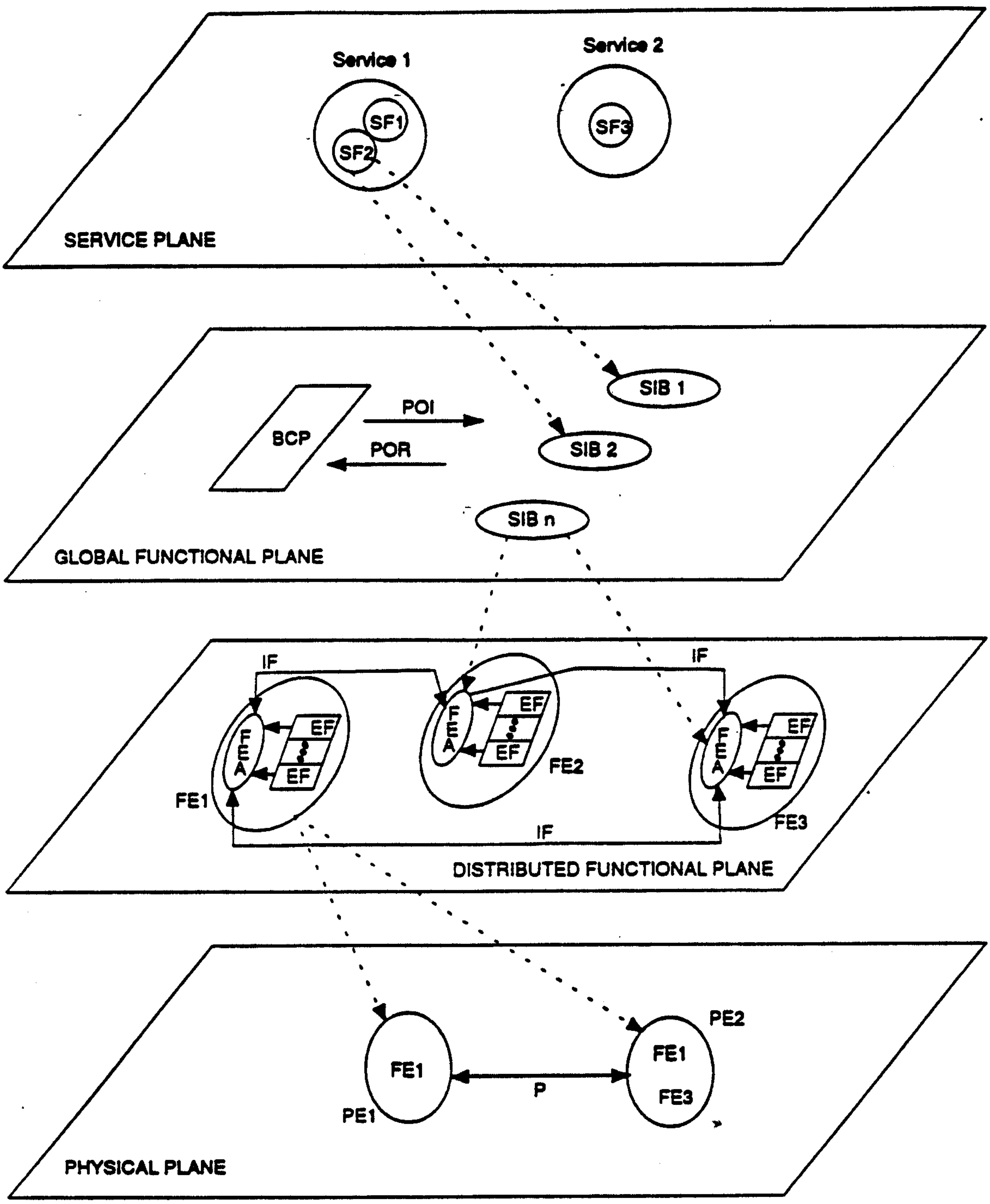
4.2 The IN Conceptual Model (INCM)

The IN Conceptual Model should not be considered in itself an architecture. It is a framework for the design and description of the IN architecture, taking into account section 1 guidelines, and in particular, the evolution of IN with its various phases.

Various "models" and "concepts" will be used in the standardization of IN. The INCM is intended to represent an integrated, formal framework within which these concepts are identified, characterized and related. It should be possible to define clearly the purpose, value and limitation of any IN concept and its relationship to other such concepts. Existing concepts may need to be adapted for use within this framework. To achieve this, the INCM consists of four "planes" where each plane represents a different abstract view of the capabilities provided by an IN-structured network. (See Figure 19). These views address service aspects, global functionality, distributed functionality and physical aspects of an IN.

4.2.1 The Service Plane

The Service Plane represents an exclusively service-oriented view. This view contains no information whatsoever regarding the implementation of the services in the network, e.g. an "IN-type" implementation is not visible. All that is perceived is the network's service-related behaviour as seen, for example, by a service user. Services are composed of one or more Service Features (SFs), which are the "lowest level" of services.



SIB: Service Independent Building Block
FE: Functional Entity
SF: Service Feature
IF: Information Flow
POI : Point Of Intitiation
POR: Point Of Return

FEA: Functional Entity Action
PE: Physical Entity
EF: Elementary Function
P: Protocol
BCP: Basic Call Process SIB
- ►Pointer

Figure 20: IN Conceptual Model

4.2.2 The Global Functional Plane

The Global Functional Plane models an IN-structured network as a single entity. Contained in this view is a global (network-wide) Basic Call Processing SIB and Point of Initiation (POI) and Point of Return (POR) between the BCP and a chain of SIBs.

4.2.3 The Distributed Functional Plane

The Distributed Functional Plane models a distributed view of an IN-structured network. Each Functional Entity (FE) may perform a variety of Functional Entity Actions (FEAs). Any given FEA may be performed within different FEs. However, a given FEA may not be distributed across FEs.

Within each FE, various FEAs may be performed by one or more EFs. The manner is which EFs result in FEAs is for further study.

Service Independent Building Blocks (SIBs) are realized in the Distributed Functional Plane (DFP) by sequence of particular FEAs performed in the FEs. Some of these FEAs result in information flows between FEs.

4.2.4 The Physical Plane

The Physical Plane models the physical aspects of IN-structured networks. The model indentifies the different physical entities and protocols that may exist in real IN-structured networks. It also indicates which FEs are implemented in which physical entities.

4.2.5 Relationship with the 3-stage method

The recommendation I.130 based 3-stage method needs enhancements for IN. The correspondence between the four planes of the IN Conceptual Model and the 3-stage method is as follows:

- Stage 1 methodology may be used to define services and service features in the Service Plane and to define SIBs in the Global Functional Plane;

- Stage 2 description methodology may be used to define the realization of SIBs in the Distributed Functional Plane;

- Protocols defined using stage 3 methodology may be applied in the Physical Plane.

4.2.6 Service Logic

Service logic may have different representations within each plane (see Figure 21), e.g.:

- Global Functional Plane: there is one set of global service logic (GSL) per Service Feature and it uses SIBs.

- Distributed Functional Plane: a set of service logic in the Global Functional Plane may be represented here by a set of Distributed Service Logic programs (DSL) co-located with the SCF FE(s).

- Physical Plane: service logic programs may be installed into and executed by any physical entity that contains the SCF FE.

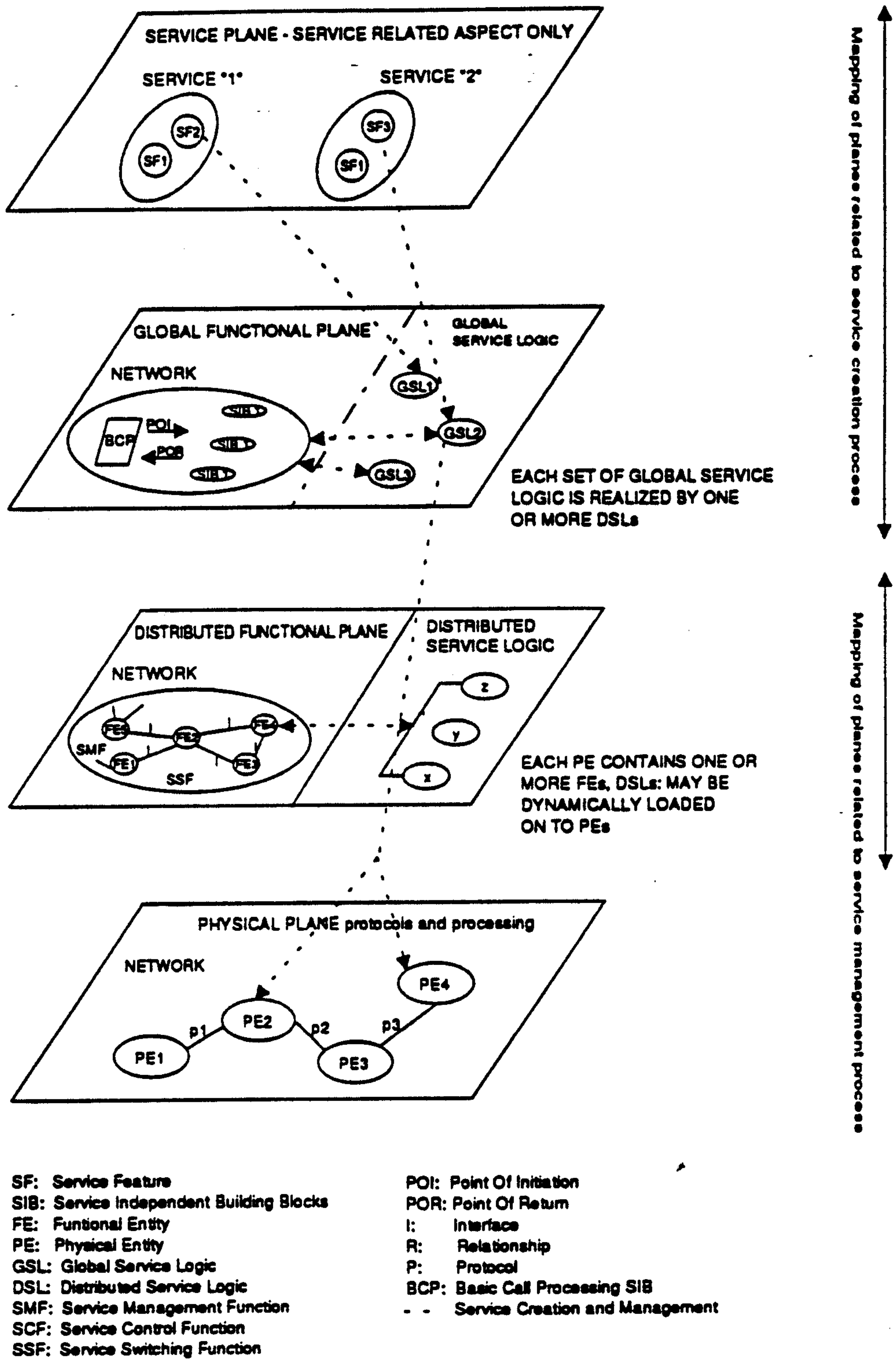


Figure 21: IN Conceptual Model with Service Logic

4.2.7 Application programming interface (API)

4.2.7.1 General API definition

An API provides a set of interfaces from an application environment to an execution environment. The execution environment provides services to the application environment.

4.2.8 Relationships among different planes

As noted in section 3.1, the entities contained in adjacent planes of the INCM are related to each other. The nature of the relationship is as follows:

- Service to GF Plane: Service Features within the Service Plane are realized in the GF plane by a combination of Global Service Logic and SIBs, including the Basic Call Process SIBs. This mapping is related to the service creation process.

- GF to Distributed Functional (DF) Plane: Each SIB identified in the GF plane must be present in at least one FE in the DF plane. A SIB may be realized in more than one FE. Thus, cooperation of several FEs may be needed. The service logic in the GF plane maps onto one or more DSLs in the DF plane. This mapping is related to the service creation process.

- DF to Physical Plane: FEs identified in the DF plane determine the behavior of the Physical Entities (PEs) onto which they are mapped. Each FE must be mapped onto one physical entity, but, each PE contains one or more FEs. Relationships between FEs, identified in the DF plane, are specified as protocols in the Physical Plane. DSLs may be dynamically loaded into physical entities and this mapping is related to the service management process.

An example of these relationships is shown in Figure 22. Here it is shown that three services (Freephone, Virtual Private Network (VPN) and Universal Personal Telecommunication (UPT) can each potentially share various service features (A, B, C, etc.). These service features may be realized in the Global Functional Plane by one or more SIBs, for example Screen and Compare.

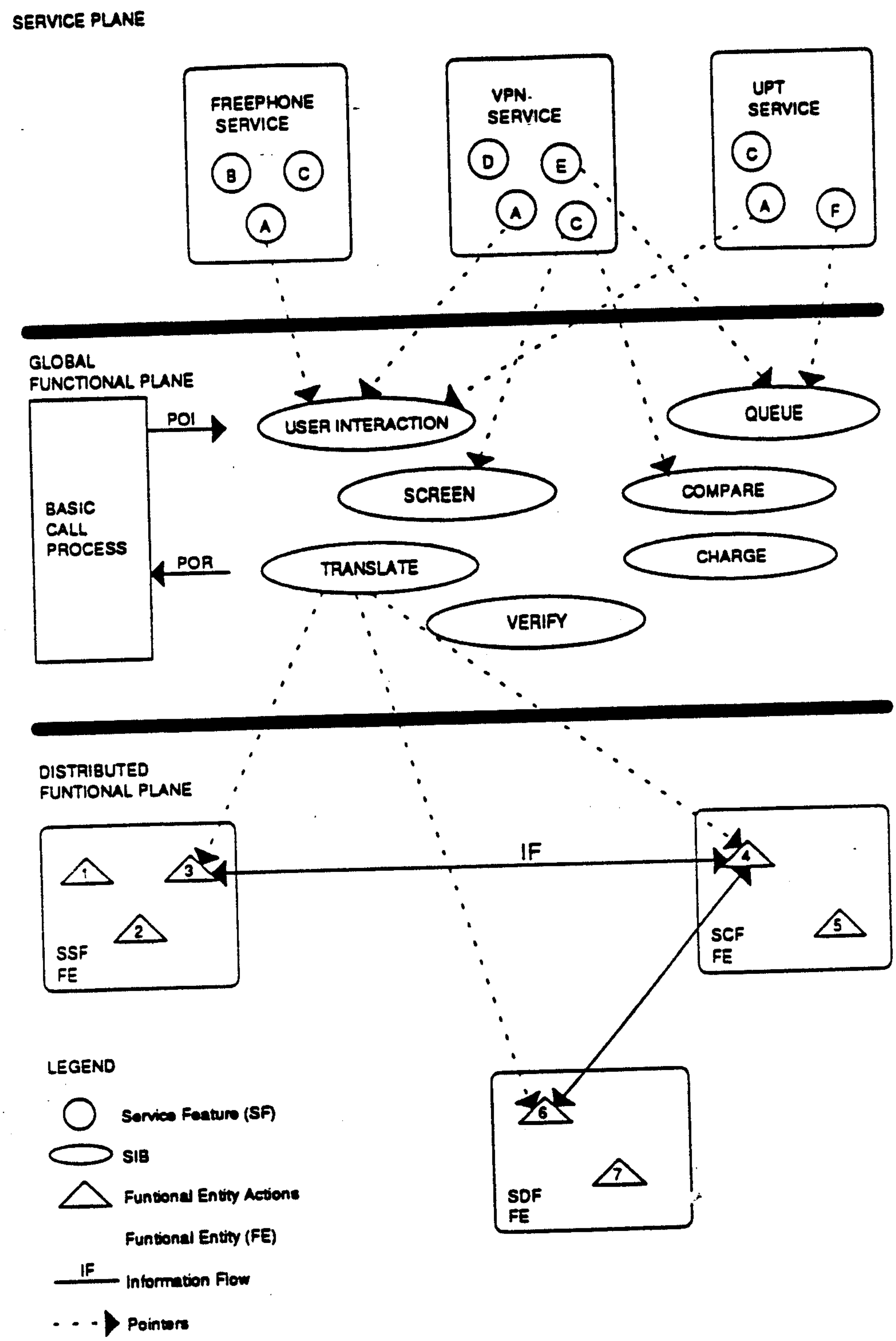


Figure 22: Service decomposition

These SIBs are realized in the Distributed Functional Plane by Functional Entity Actions (FEAs) (e.g. 1, 2, 3, 4) which occur within one or more FEs, e.g. Service Switching Function (SSF), Service Control Function (SCF) and Service Data Function (SDF). When more than one FE is required, information flows result between FEAs as shown.

4.2.9 Service interaction

A set of rules must be provided at the service plane level - these services may be IN or non-IN. These interactions will also impact all the other planes. In addition to the set of rules at the service plane level, a robust mechanism for resolving easy feature interaction should be provided in other planes in a service-independent manner to facilitate rapid feature introduction.

4.2.10 Service and network interworking

Service interworking (for further study)
Network interworking

1) Network interworking on the distributed functional plane:

- this plane should be explicitly divided into several parts, each of which represent one functional network;

- network interworking requires that relationships are defined between pairs of functional entities in different functional networks (e.g., between an SCF in network A and an SDF in network B);

- each network interworking interaction between communicating pairs of functional entities is termed an information flow. The network interworking relationship between any pair of functional entities is the set of network interworking related information flows between them;

- the semantic meaning and information content of each information flow needs to consider network interworking capabilities, network security and network integrity;

- Functional entities which support internetworking provide internetworking functionality based on the associated information flows and functional entity actions.

2) Network interworking in the physical plane:

- this plane should be explicitly divided into several parts, each of which represent one physical network;

- functional entities are allocated to physical entities in each of the networks that interwork;

- network interworking protocols are defined and standardized, to support the network interworking relationship between two functional entities, each of which is located in a different physical network.

4.2.11 Management functionality

Management is related to all planes of the IN Conceptual Model. In an IN-structured network there is a need to consider both the service and network aspects of management. Specific text on these aspects is contained within each architectural section (sections 4 to 7).

One particular aspect of management, (e.g. service creation, service introduction, service tailoring, customer control etc.) can be viewed as non real time, and independent of the actual real-time service execution. In a multivendor environment, different versions of physical representations of the same functional entity may exist, where versions may contain subsets of capabilities of other versions.

4.3 Service plane architecture

4.3.1 General

The service plane illustrates that IN-supported services can be described to the end user or subscriber by means of a set of generic blocks called "Service Features".

A Service is a stand-alone commercial offering, characterized by one or more core service features, and can be optionally enhanced by other service features.

A Service Feature is a specific aspect of a service that can also be used in conjunction with other services/service features as part of commercial offering. It is either a core part of a service or an optional part offered as an enhancement to a service.

The service plane represents an exclusively service-oriented view. This view contains no information whatsoever regarding the implementation of the services in the network (for instance, an IN type of implementation is invisible). All that is perceived is the network's service-related behaviour as seen, for example, by a service user.

Furthermore, Management Services are contained in the Service Plane; they can be described to the end user by means of Service Management Features.

4.3.2 Characterization of services and service capability requirements

Characterization of services and service features is to identify service independent capabilities that are required to construct and/or customize services by the users or network operators. Examples of service capabilities required from the user point of view are, call queueing, customized announcement, etc.

There is a need for a structured approach with which to classify service characteristics and identify service capabilities. The structured approach shown in Figure 23 below demonstrates a methodology for analyzing services and decomposing services into service independent building blocks (SIBs). These reusable service independent building blocks (such as Translation, User interaction, or charging) will form the basis for input to global functional plane modelling and distributed functional plane modelling.

It is recommended that activities involving functional modelling make use of the results of such service analysis, based on the characterization of services for verification of their models, and to ensure a unified model for service processing.

4.3.3 Service and service feature interaction

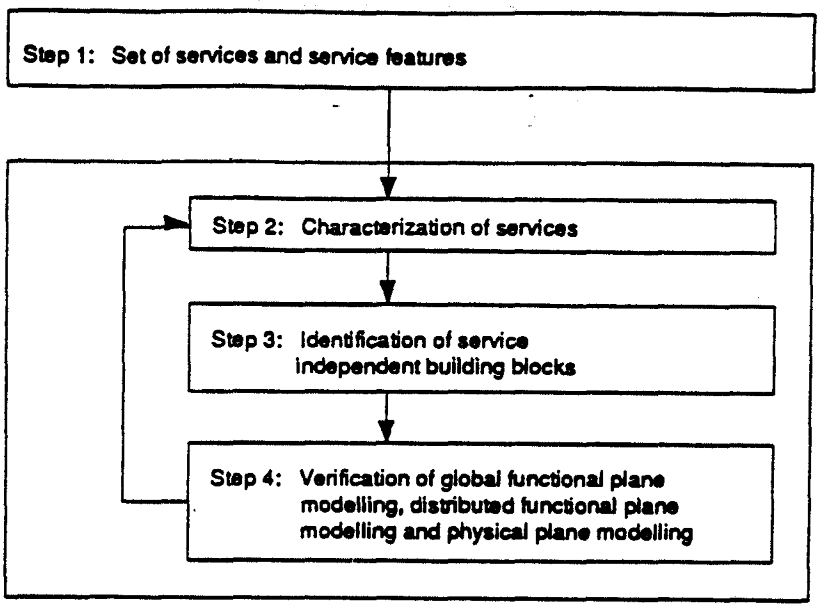
This section focuses on the interaction between IN supported services and other supplementary services and not between basic services. The service interactions are described from the customer and user point of view.

Service interaction applies to all interactions of the service being described with other services which have been already identified.

Service feature interactions may occur (for example):

1) among different features associated with the same service;

2) between features associated with a service for a given service-user and features associated with other services the same user may have requested or been assigned;



Step 1: Existing service descriptions (Stage 1 Service Description) as well as emerging service descriptions are selected as candidates for analysis.

Step 2: These services are characterized based on the principle of decomposition of services into functions.

Step 3: The analysis in step 2 results in requirements in the form of service independent building blocks as the input to Global Functional Plane Modelling, Distributed Functional Plane Modelling and Physical Plane Modelling.

Step 4: Verification of the Global Functional Plane Modelling, Distributed Functional Plane Modelling and Physical Plane Modelling results in improved SIBs through feedback to step 2.

Figure 23: Methodology for identifying SIBs

3) between features associated with a service for a given service-user and features associated with possible services related to the terminal/calling line that the user is currently using, e.g. in case terminal mobility and/or personnel mobility are involved.

An IN structured network handles multiple services for the same call. The necessary interactions shall be defined for the processing of several services for the same call. When multiple services can be activated concurrently, some prioritization of services will be necessary. User specific requests may take priority over a group service request. Additionally, certain services may override or deactivate other services.

The service interaction is part of the specification of services, and should be dealt with in the service plane modelling. There are often many ways to deal with an interaction between two or more services. In an IN structured network, service interactions may be customized.

How service interactions are implemented is not visible in the service plane. The usage of the service independent mechanism in the IN architecture to handle service interactions will be visible in the Global Functional, the Distributed Functional and the Physical Planes.

The following issues need consideration when service interactions are specified:

- At different phases of a call, i.e., originating, terminating, interrupt (active) and release phase of call processing.
- When a service spans over more than one network. They may impose additional requirements on service interaction, which need further study.
- Service interaction may occur between services offered to a single user, as well as between services offered to different interacting users.

Example of service interactions are given below:

- Abbreviated Dialling and Number Screening
- Freephone and Call Forwarding Unconditional
- CLIR and CLIP (Calling line Identification Restriction and Presentation)
- Call Forwarding and Premium Rate Service
- Call Waiting and Call Forwarding Busy
- Conference Call and CUG (Closed User Group)
- Meet ME Conference and CUG

Examples of different interactions between freephone and call forwarding unconditional are:

1) Freephone calls shall not be forwarded, but the selected destinations shall be called.
2) Freephone calls shall be forwarded like other terminating calls.
3) A freephone destination, which has activated call forwarding unconditional, shall not be selected for freephone calls.

4.3.4 Service plane modelling

Services are comprised of one or more service features (SF). A Service Feature is the smallest part of a service that can be perceived by the service user. These SFs can also be used as building blocks in the specification and design of new, more complex services. SFs are comprised of one or more SIBs which are described in Section 4.4.

All individual telecommunications services identified in the service plane should be described as seen from the user's viewpoint without reference to how the services are implemented in the network.

In the service plane architecture, it is stressed that all capabilities experienced by a service user of the network represent telecommunication services (basic or supplementary). The service user may make use of the service for his own communication needs or may combine a number of services together and with perhaps additional capabilities, use the combination as a means of providing communications to other (third) parties.

4.4 Global Functional Plane Architecture

4.4.1 General

The concepts for the Intelligent Network are embodied in the IN Conceptual Model (INCM) as described in associated Section 4.2 and 4.3. This section describes the Global Functional Plane (GFP) of the INCM with respect to the composition of the plane, and its relationship to adjacent planes. This plane is viewed as the proper location for the modular functionality from which services are to be constructed.

The Global Functional Plane models network functionality from a global, or network-wide, point of view. As such, the IN Structured Network is said to be viewed as a single entity in the GFP. In this plane, Services and Service Features are redefined in terms of the broad network functions required to support them. These functions are neither Service or Service Feature (SF) specific and are referred to as Service Independent Building Blocks (SIBs).

The Global Functional Plane is located between the Service Plane and the Distributed Functional Plane as illustrated in Figure 24. Services identified in the Service Plane are decomposed into their Service Features then mapped onto one or more SIBs in the GFP. Each SIBs is similarly mapped onto one or more Functional Entities in the Distributed Functional Plane.

Contained in the Global Functional Plane are (refer to Figure 24):

- Basic Call Process (BCP) SIB which identifies the normal call process from which IN services are launched, including Points of Initiation (POI) and Points of Return (POR) which provide the interface from the BCP to Global Service Logic.

- SIBs which are standard reuseable network wide capabilities used to realise services and SFs

- Global Service Logic (GSL) which described how SIBs are chained together to describe Service Features. The GSL also describes interaction between the BCP and the SIB chains.

4.4.2 Global Functional Plane modelling

By definition, SIBs, including the BCP, are service independent and cannot contain knowledge of subsequent SIBs. Therefore, Global Service Logic (GSL) is the only element in the GFP which is specifically service dependent.

Referring the illustration in Figure 25 in the GFP, normal or non-IN supported services are processed within the BCP. When an IN supported service is to be invoked, its GSL is launched at Point of Initiation (POI) by a triggering mechanism from the BCP.

In order to chain SIBs together, knowledge of the connection pattern, decision options, and data required by SIBs must be available. Therefore, the pattern of how SIBs are chained together must be maintained within the GFP, and described in the GSL. The GSL describes subsequential SIB chaining, potential branching, and where branches rejoin.

At the end of the chain of SIBs, the GSL also describes the returning point to the BCP by indicating the specific Point of Return (POR).

For a given service/SF, at least one POI is required. However, depending upon the logic required to support the service/SF, multiple PORs may be defined.

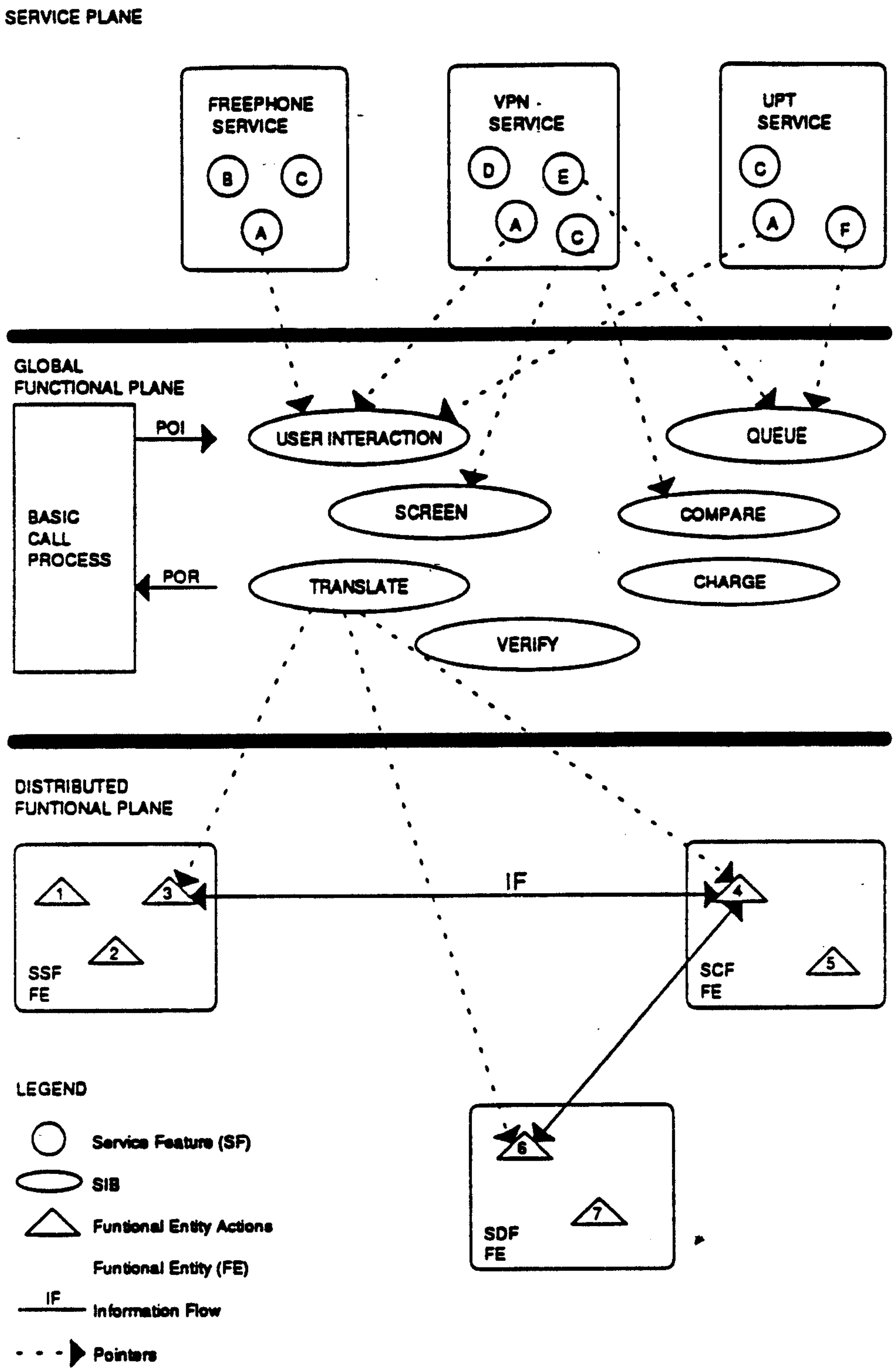


Figure 24: Service decomposition

The process of how the GSL is described through the Service Creation Environment using the Application Programming Interface is an area for further study.

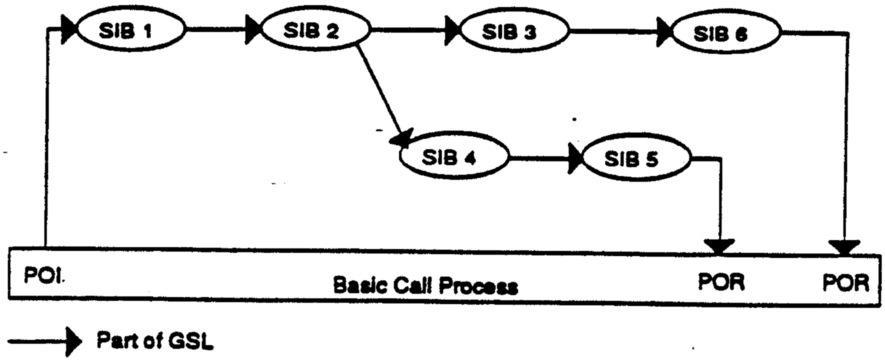


Figure 25: Modelling of Global Functional Plane

4.4.3 Service independent building blocks

4.4.3.1 Definition of a SIB

A SIB is a standard reusable network-wide capability residing in the Global Functional Plane used to create Service Features. SIBs are of a global nature and their detailed realization is not considered at this level but can be found in the Distributed Functional Plane and the Physical Plane. The SIBs are reuseable and can be chained together in various combinations to realise services and SFs in the service plane. SIBs are defined to be independent of the specific service and technology for which or on which they will be realized.

4.4.3.2 Characteristics of a SIB

SIBs have the following characteristics (note):

- SIBs are defined completely independent from consideration of any specific distributed functional and physical plane architectures (network implementation independent).
- Each SIB should have a unified and stable interface.
- DFP interaction among FEs is transparent to the SIBs in the GFP.
- Individual SIBs must be defined using a standard methodology to allow:
  - multi-vendor IN products to identically support them;
  - service designers to have a common understanding of the SIB.

NOTE: No implication of importance is meant to be implied by the order:

- SIBs are the monolithic building blocks (their detailed implementation is hidden) that the service designer will use to develop new services.
- All Service Features (SFs) are described by one SIB or a chain of SIBs.
- All SFs can be defined by a finite number of SIBs.
- A SIB defines one complete activity.
- SIBs are realized in the DFP by Function Entity Actions which may reside in one or more Functional Entities (FEs).
- A SIB has one logical starting point and one or more logical end points. Data required by each SIB is defined by SIB Support Data parameters and Call Instance Data parameters.
- SIBs are global in nature and their locations need not be considered as the whole network is regarded as a single entity in the GFP.
- SIBs are reusable. They are used without modification for other services.

4.4.3.3 Data Parameters for SIBs

By definition, SIBs are independent of the service/SF they are used to represent. They have no knowledge about previous or subsequent SIBs which are used to describe the service feature.

In order to describe service features with these generic SIBs, some elements of service dependence is needed. Service dependence can be described using data parameters which enables a SIB to be tailored to perform the desired functionality. Data parameters are specified independently for each SIB and are made available to the SIB through Global Service Logic.

Two types of data parameters are required for each SIB, dynamic parameters called Call Instance Data (CID) and static parameters called Service Support Data (SSD).

4.4.3.3.1 Call Instance Data (CID)

Call Instance Data defines dynamic parameters whose value will change with each call instance. They are used to specify subscriber specific details like calling or called line information. This data can be:

- made available from the BCP SIB, (e.g. Calling Line Identification);
- generated by a SIB, (e.g. a translated number); or,
- entered by the subscriber, (e.g. a dialed number or a PIN code).

Associated with each CID value is a logical name which is referred to as the CID Field Pointer, (CIDFP). If a SIB requires CID to perform its function, there will be an associated CIDFP assigned through SSD, (refer to Section 4.4.3.3.2). For instance, the Translate SIB's CID which defines what is to be translated, is called *Information*. The Translate SIB's SSD parameter which defines where this data can be found is *CIDFP-Info*.

Since the CID value can vary with each call instance, service features can be written with data flexibility. In the above Translate SIB example, one service feature may require translation of a calling number, while another service feature will require translation of the called number. In both cases, the data required by the SIB is specified by the *Information* CID, but the *CIDFP- Info* changes. In the first service feature the value of *CIDFP- Info* is set to CLI, while the second service feature sets the value of *CIDFP- Info* to CLD.

Once a CIDFP have been specified for a service feature, it can be referenced by subsequent SIBs and the CID value can be made available to all subsequent SIBs in the SIB chain. This CIDFP is said to be fixed for that service and is constant for all instances of that service. The actual value of the CID changes for each call instance of that service feature.

4.4.3.3.2 Service Support Data (SSD)

Service Support Data defines data parameters required by a SIB which are specific to the service feature description. When a SIB is included in the GSL of a service description the GSL will specify the SSD values for the SIB. SSD consists of:

i) Fixed Parameters
These are data parameters whose values are fixed for all call instances. For instance, the "*File Indicator*" SSD for the Translate SIB needs to be specified uniquely for each occurrence of that SIB in a given service feature. The "*File Indicator*" SSD value is then ~~said to~~ be fixed as it's value is determined by the service/SF description, not by the call instance.

    If a service/SF is described using multiple occurrences of the same SIB, then fixed SSD parameters are defined uniquely for each occurrence.

ii) Field Pointers
Field Pointers identify which CID is required by the SIB, and in doing so provide a logical location for that data. They are signified by "*CIDFP-xxxx*", where "*xxxx*" names the data required. For instance, "*CIDFP-Info*" for the Translate SIB will specify which CID element is to be translated.

    If more than one CID is required by a SIB to perform it's function, then the SSD data parameters will contain multiple Field Pointers.

4.4.3.4 Method to describe SIBs

The SIBs provide the modularity within the Global Functional Plane that is required by the definition and objectives of the IN concept. In order to effectively progress such studies a method is required to characterize and technically describe the SIBs.

Techniques analogous to those used in the 3-stage service definition methodology (Recommendation I.130), i.e. prose description, static description, and dynamic description, are appropriate.

The procedure outlined in Figure 27 can be used to determine if new SIBs are required to support new services.

The following terms are used in the SIBs identification method:

4.4.3.4.1 Definition

Prose description of the SIB from the service creation point of view.

4.4.3.4.2 Operation

Description of actions performed by the SIB. The operations section expands on the definition, to allow the reader to clearly understand the operation that this SIB is intended to perform.

4.4.3.4.3 Potential Service Applications

Service examples of where this SIB can be used.

4.4.3.4.4 Input

Input to each SIB is specified as three distinct elements:

- one logical starting point
- Service Support Data which defines parameters which are specified by the service description
- Call Instance Data which are specific to that call instance

4.4.3.4.5 Output

Output from each SIB is specified as two distinct elements:

- one or more logical end points
- Call Instance Data which defines data parameters specific to that call instance which results from the execution of that SIB and are required by other SIBs or the BCP to complete the call service instance

4.4.3.4.6 Graphic representation

A graphic representation describing input, operations and output of the SIB and is illustrated in Figure 26. Each SIB is characterized by having one logical input and one or more logical outputs. These logic flows are shown by the solid arrows on the left and right of the diagram. Each logic flow is specified, above each arrow. SSD parameters are identified by the dashed arrow at the top of the diagram and are specified beside the dashed arrow. Similarly, CID parameters are specified below the diagram. Input CID parameters are separated fro the output parameters.

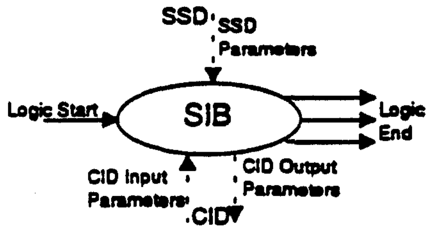


Figure 26

4.4.3.4.7 SDL diagram

Graphic representation of the stage 1 description of the SIB using SDL macro diagrams (Recommendation Z.100).

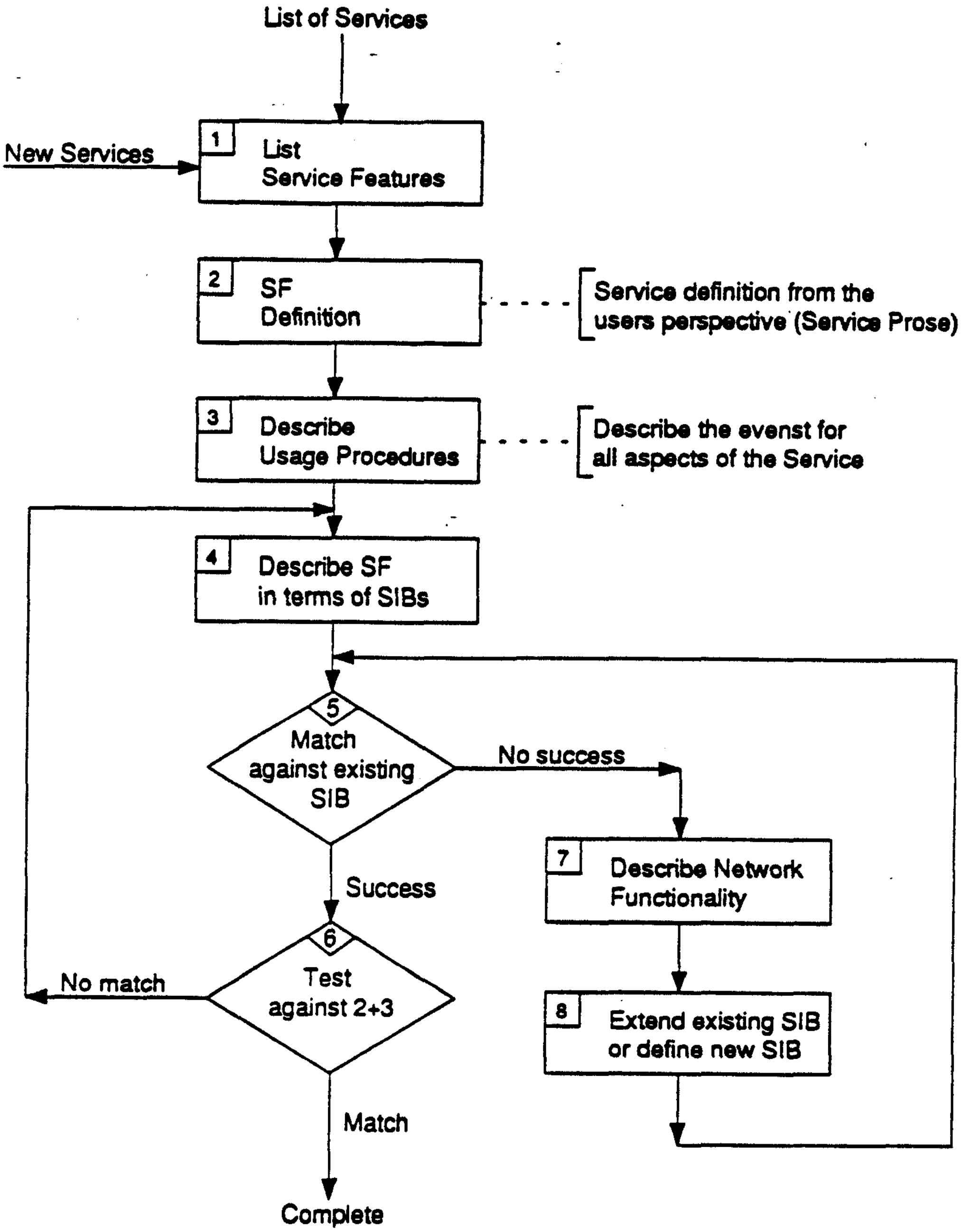


Figure 27: Flowchart to identify Service Independent Building Blocks

4.4.3.5 Flowchart analysis

The starting point for the determination of SIBs is services. In the Service Plane of the IN Conceptual Model (INCM), services are decomposed into their Service Features (SF), which are the features that comprise the service. Full service descriptions must be available for the new service being analyzed prior to identifying SIBs.

Given that a catalogue of Services, SFs, and SIBs exist, the following description explains how analysis of a new service may lead to the identification of new SIBs (refer to Figure 27).

1) List Service Features

   Decompose the new service into its SFs.

2) Service Feature Definition

   Define each SF by describing the service provided from the end users (Subscriber) perspective. This definition is referred to as the Service Prose. Information should be available from the Stage 1 Service Description.

3) Describe Usage Procedure

   Describe the chain of events seen by the user for this SF. This includes service subscription, activation, modification, and call scenarios for the SF.

4) Describe Service Feature in terms of SIB

   Describe Service Feature in terms of the modular network functions represented by SIBs.

5) ~~Map to existing~~ SIBs

   Compare the above (Steps 3 and 4), with the characteristic lists for established SIBs.

6) Test against 2 and 3

   Verify the robustness of the SF by analyzing the SDL representation with the SF Definition and Usage Procedure (from Steps 2 and 3). Failure to pass this verification indicates that the analysis in Steps 4 and 5 was incorrect or incomplete.

7) Describe Additional Network Functionality Required

   Describe what functions must be provided by the network, in addition to those of existing SIBs, to fully support the SF.

8) Extend existing SIB, or define new SIB

   If possible, extend the capabilities of an existing SIB (e.g. additional "type") to provide the additional functionality required to support the SF. If such extension is not possible, then define a new SIB. Complete the definition of the extended or new SIB by providing the detailed information in section 3.4.

4.4.4 Basic call process

4.4.4.1 General

The Basic Call Process (BCP) is responsible for providing basic call connectivity between parties in the network. The BCP can be viewed as a specialized SIB which then provides basic (e.g. Q.17) call capabilities, including:

- connecting call, with appropriate disposition;
- disconnecting calls, with appropriate disposition;
- retaining CID for further processing of that call instance.

4.4.4.2 Basic Call Process Functionality

IN supported services/SFs are represented through the use of chains of SIBs connected to the BCP SIB. The interface points between the BCP SIB and the chains of SIBs are described as Points of Initiation, and Points of Return, with the following definitions:

i) A Point of Initiation is the BCP functional launching point for the SIB chains.

ii) A Point of Return identifies the functional point in the BCP where the of SIB chains terminate.

A graphical illustration of the POI/POR/BCP functionality is shown if Figure 28. The number and location of these points must be determined by analysis of the capabilities required for future Capability Sets.

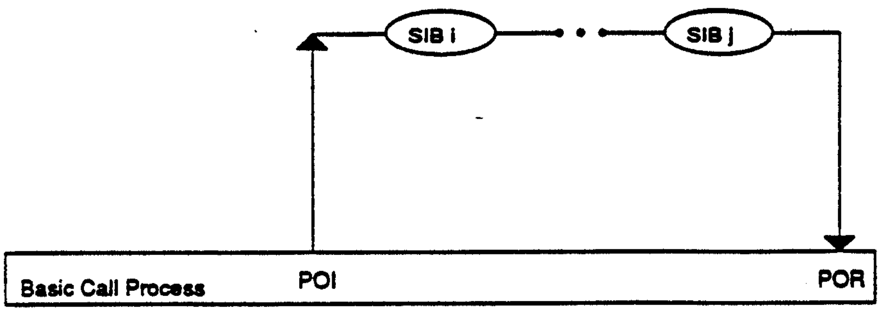


Figure 28: POI/POR/BCP Relationship

The need for specific POI/POR functionality is that the same chain of SIBs may represent a different service if launched from a different point in the BCP. Similarly, the same chain of SIBs launched from the same point may represent a different service if returned to the BCP at a different point.

4.4.5 Global Service Logic

This section represents the role of the Global Service Logic (GSL) in the Global Functional Plane.

4.4.5.1 General

The GSL can be defined as the "glue" that defines the order in which SIBs will be chained together to accomplish services. Each instance of Global Service Logic is (potentially) unique to each individual call, but uses common elements, comprising specifically:

- BCP interaction points (POI and POR);
- SIBs;
- logical connections between SIBs, and between SIBs and BCP interaction points;
- Input and Output data parameters, Service Support Data and Call Instance Data defined for each SIB;

Based upon the functionality of these common elements, Global Service Logic will "chain together" these elements to provide a specific service.

In order to more completely demonstrate how GSL operates, a generic example of a service is illustrated in Figure 29. This diagram shows that specific SIB chains launched from designated POI, are activated in a particular order and are returned to the appropriate PORs, as required by the GSL. To aviod complexity, the SIB data parameters are not shown.

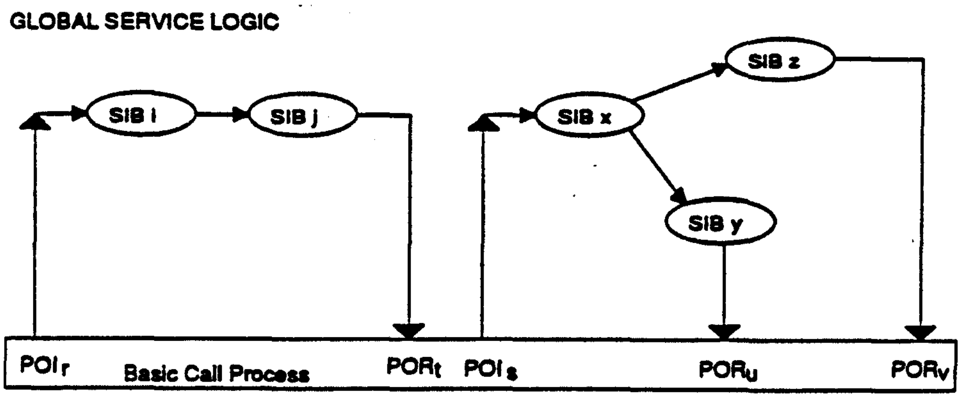


Figure 29: GSL Example

4.4.5.2 Relationship between GSL and BCP

Global Service Logic on the global functional plane views Basic Call Process as a single resource. Based upon such a view of IN services, the following are identified as necessary interactions between global service logic and BCP, for example:

Communications form BCP to GSL:

i) Logical start for SIB chains - which is represented by POIs;

ii) Data - which is represented by Call Instance Data, which is required by SIB chains for processing IN service features. Example of specific Call Instance Data which the BCP may be responsible for could include Calling Line Identity and Dialled Number.

Communications from Global Service Logic to BCP:

i) Logical termination for SIB chains - which is represented by PORs;

ii) Data - which is represented by Call Instance Data that have been defined by one or more SIBs on a SIB chain. An example of such a Call Instance Data could be a Destination Number. GSL ensures that all relevant CID is maintained throughout multiple SIB chains until termination of each call instance.

4.4.5.3 Relationship between Global Service Logic and SIBs

The remaining components of GSL needed to define a service/SF are the collection of SIBs (including their Service Support and Call Instance Data), and the topology of their interconnection (to each other and to the POIs and PORs of the BCP). This specifices the functionality required to support the service/SF and the sequence of occurrence of this functionality.

4.4.6 Mapping of the Service Plane to the Global Functional Plane

Referring to Figure 30 in the GFP, or non-IN services are processed through the Basic Call Process. When an IN Service Feature (SF) is to be invoked, it is initiated by a triggering mechanism from the Basic Call Process. The chain "link" pattern which describes the SF must then be obtained by the Global Service Logic in order to process the SF. As new SFs are designed their SIB descriptions must be made available to the Global Service Logic.

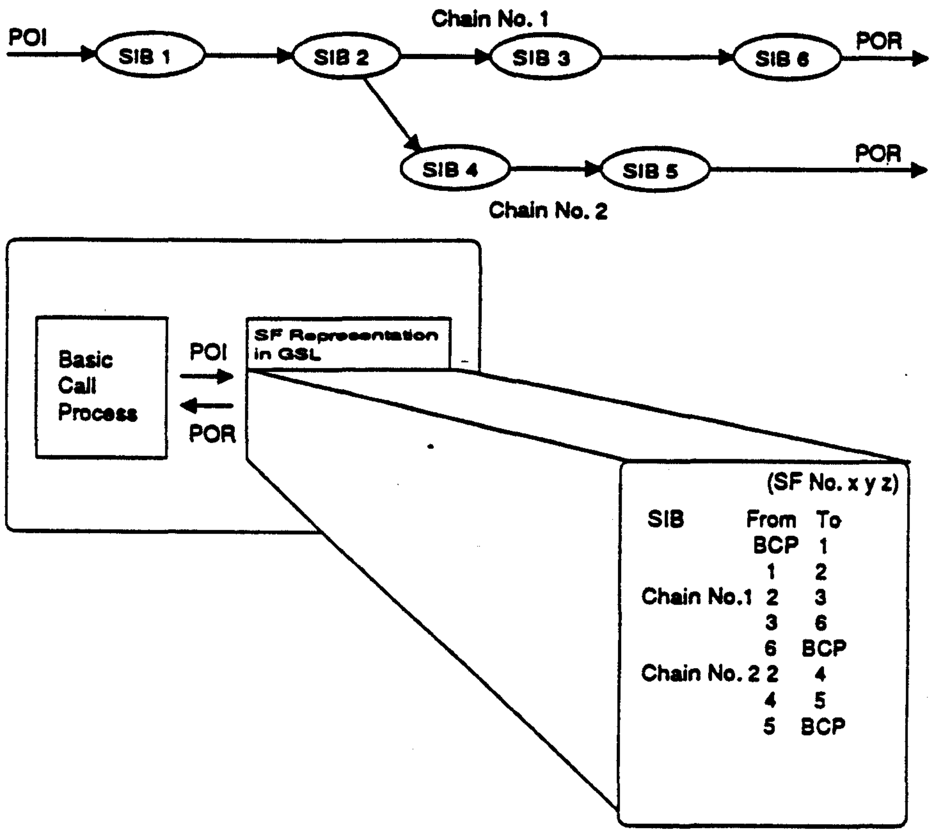


Figure 30: Global Funtional Plane Blueprint of Service Feature

4.5 Distributed Functional Plane Architecture

4.5.1 General

The requirements and assumptions for the IN Distributed Functional Plane (DFP) architecture are as follows:

a) the DFP architecture is consistent with the framework defined by the IN Conceptual Model:

- it identifies the specific elements and the relationship between them that are necessary to support the objectives of IN.

- the DFP architecture models the functions to be performed in IN structured networks.

b) the DFP architecture provides the flexibility to support a large variety of services and facilitates the evolution of IN by organizing the functional capabilities in an open-ended and modular structure to achieve service independence.

c) the DFP architecture is vendor/implementation independent, thereby providing the flexibility for multiple physical networking configuration and placing no constraints on national network architecture beyond the network and interface standards which will be developed for IN structured networks.

d) the definition of the DFP architecture initially accommodates service execution capabilities and will accommodate service creation and service and network management capabilities when they become available.

4.5.2 Distributed Funtional Plane Model

Figure 31 identifies the IN DFP model.

4.5.2.1 Explanation of Diagram

4.5.2.1.1 Functional Entities

A Functional Entity (FE) is a grouping of functions in a single location and a subset of the total set of functions required to provide a service. One or more functional entities can be located in the same Physical Entity. Different functional entities contain different functions, and may also contain one or more of the same functions. In addition, one functional entity cannot be split between two physical entities; the functional entity is mapped entirely within a single physical entity. Finally, duplicate instances of a functional entity can be mapped to different physical entities, though not to the same physical entity.

FEs are represented by ovals in the functional model diagram.

FEs are assigned unique FUNCTIONAL ENTITY IDENTIFIERS (eg. CCF, for Call Control Function).

FE descriptions do not include utility/housekeeping functions which are not directly involved in providing a service. This explains why there is no FE identified to describe the communication between FEs.

The physical location of function is not the only criteria for grouping functions into an FE. FE grouping criteria should take into consideration all technological and business requirements.

4.5.2.1.2 Relationships

Each interaction between a communicating pair of FEs in a model is termed an "Information Flow". The relationship between any communicating pair of FEs in the model is defined by a set of information flows.

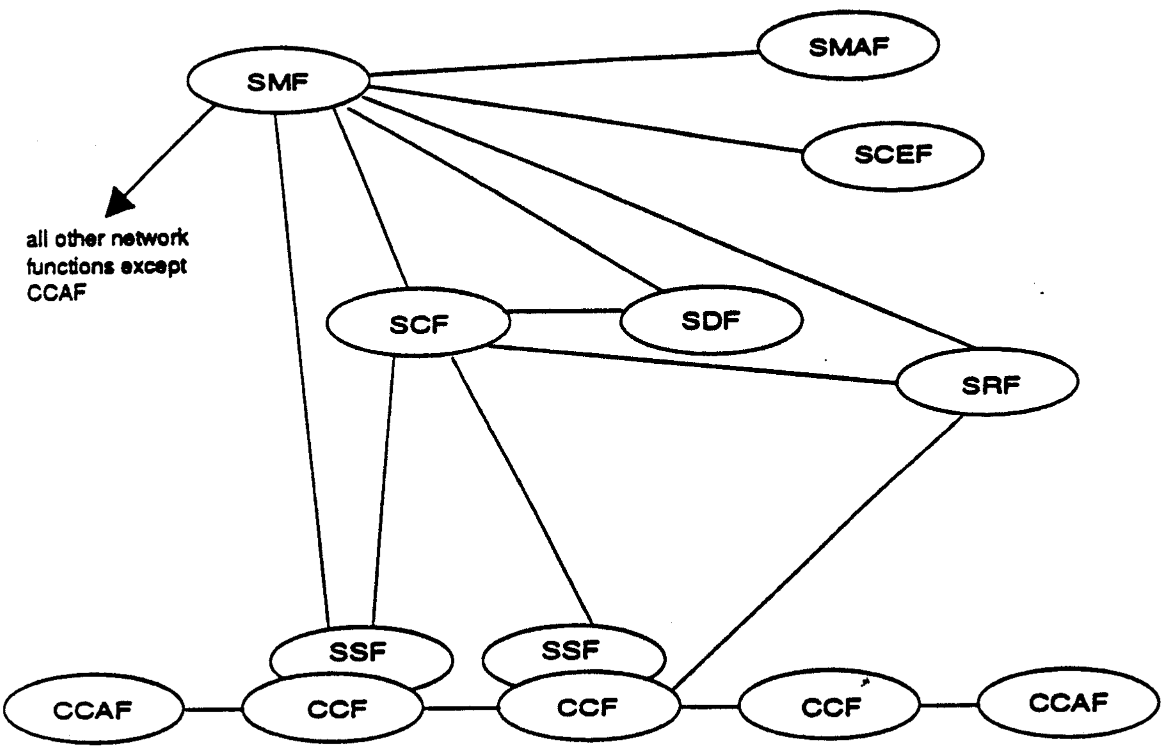
FE Relationships are the lines between FEs in the functional model diagram.

Relationships may have TYPE identifiers assigned, which uniquely identify specific sets of information flows within the model (eg. r1, r2, etc). The same relationship type may occur more than once in a functional model.

If the model does not show a line between FEs, there is no identified relationship which requires standardization between them.

If a communicating pair of FEs is located in physically separate entities, the relationship between them defines the information transfer requirements for a protocol between the physical entities.

4.5.2.2 IN Functional Model



CCF	Call Control Function	SDF	Service Data Function
SSF	Service Switching Function	SMF	Service Management Function
SCF	Service Control Function	SCEF	Service Creation Environment Function
CCAF	Call Control Agent Function	SMAF	Service Management Agent Function
SRF	Specialized Resource Function		

Figure 31: IN Distributed Functional Plane Model

Note: The SSF/CCF have identical functionality and are only shown for some procedures like assist.

Note: The definition of CCAF and CCF are based on corresponding Q.71 ISDN definitions, but may be modified for use in IN.

4.5.2.3 Definition of Functional Entities Related to IN Service Execution

THE CCA FUNCTION (CCAF)

The CCAF is the call control agent (CCA) function that provides access for users. It is the interface between user and network call control functions. It:

a) provides for user access, interacting with the user to establish, maintain, modify and release, as required, a call or instance of service.

b) accesses the service-providing capabilities of the Call Control Function (CCF), using service requests (e.g. setup, transfer, hold, etc.) for the establishment, manipulation and release of a call or instance of service.

c) receives indications relating to the call or service from the CCF entity and relays them to the user as required.

d) maintains call/service state information as perceived by this functional entity.

THE CC FUNCTION (CCF)

The CCF is the call control (CC) function in the network that provides call/service processing and control. It:

a) establishes, manipulates and releases call/connection instances as "requested" by the CCAF.

b) provides the capability to associate and relate CCAF entities that are involved in a particular call and/or connection instance. (That may be on SSF requests.)

c) manages the relationship between CCAF functional entities involved in a call (e.g. supervises the overall perspective of the call and/or connection instance).

d) provides trigger mechanisms to access IN functionality (e.g., passes events to the SSF).

e) is managed, updated and/or otherwise administered for its IN-related functions (i.e., tigger mechanisms) by a Service Management Function (SMF).

THE SS FUNCTION (SSF)

The SSF is the service switching (SS) function, which, associated with the CCF, provides the set of functions required for interaction between the CCF and a Service Control Function (SCF). It:

a) extends the logic of the CCF to include recognition of service control triggers and to interact with the Service Control Function (SCF).

b) manages signaling between the CCF and the SCF.

c) modifies call/connection processing functions (in the CCF) as required to process requests for IN provided service usage under the control of the SCF.

d) is managed, updated and/or otherwise administered by a Service Management Function (SMF).

THE SC FUNCTION (SCF)

The Service Control Function (SCF) is a function that commands call control functions in the processing of IN provided and/or custom service requests. The SCF may interact with other entities to access additional logic or to obtain information (service or user data) required to process a call/service instance. It:

a) interfaces and interacts with SSF/CCF, Specialized Resource Function (SRF) and Service Data Function (SDF) functional entities.

b) contains the logic and processing capability required to handle IN provided service attempts.

c) interfaces and interacts with other SCFs, if necessary.

d) is managed, updated and/or otherwise administered by a SMF.

THE SD FUNCTION (SDF)

The Service Data Function (SDF) contains customer and network data for real time access by the SCF in the execution of an IN provided service. It:

a) interfaces and interacts with SCsF as required.

b) interfaces and interacts with other SDFs, if necessary.

c) is managed, updated and/or otherwise administered by an SMF.

Note: The SDF contains data relating directly to the provision or operation of IN provided services. Thus it does not necessarily encompass data provided by third party such as credit information, but may provide access to these data.

THE SR FUNCTION (SRF)

The Specialized Resource Function (SRF) provides the specialized resources required for the execution of IN provided services (e.g., digit receivers, announcements, conference bridges, etc). It:

a) interfaces and interacts with SCF and SSF (and with the CCF).

b) is managed, updated and/or otherwise administered by a SMF.

c) may contain the logic and processing capability to receive/send and convert information received from users.

d) may contain functionality similar to the CCF to manage bearer connections to the specialized resources.

4.5.2.4 Definition of IN Service Creation/Management Related Functional Entities

SERVICE CREATION ENVIRONMENT FUNCTION (SCEF)

This function allows services provided in intelligent network to be defined, developed, tested and input to SMF. Output of this function involves among other service logic, service data template and service trigger information.

SERVICE MANAGEMENT AGENT FUNCTION (SMAF)

This function provides an interface between service managers and the SMF.

It allows service managers to manage their services (through access to the SMF).

SERVICE MANAGEMENT FUNCTION (SMF)

This function allows deployment and provision of IN provided services and allows the support of ongoing operation.

Particularly, for a given service, it allows to coordinate different SCF and SDF instances, e.g. :

- billing and statistic information are received from the SCFs, and made available to authorized service managers through the SMAF;
- modifications in service data are distributed in SDFs, and it keeps track of the reference service data values.

The SMF manages, updates and/or administers service related information in SRF, SSF and CCF.

Note: The relation between TMN (see Recommendation M.30), SMAF and SMF is for further study.

4.5.3 Functional Entity Call/Service Processing Models

4.5.3.1 General

IN call/service logic processing encompasses call and connection processing in the SSF/CCF, service logic execution in the SCF, and the use of supporting resource and data in the SRF and SDF, respectively. This section describes this IN call/service logic processing in term of call modelling and modelling of service logic processing.

- Call modelling provides a high-level service and vendor/implementation independent abstract of IN call and connection processing in the SSF/CCF. This abstraction provides an observable view of SSF/CCF activities and resources to the SCF, enabling the SCF can interact with the SSF in the course of executing service logic.
- The modelling of service logic processing provides an abstraction of SCF activities and resources needed to support this service logic execution, as well as an abstraction of SRF and SDF activities and resources accessible to the SCF.

Since call modelling and modelling of service logic processing only provides an observable (i.e., external) view of SSF/CCF, SCF, SRF, and SDF activities and resources, this modelling does not imply an obligation to vendors to implement functional entities into products as a one-to-one mapping of FE model components.

4.5.3.2 Modelling Objectives/Criteria

4.5.3.2.1 Call Modelling Objectives/Criteria

The general call modelling objectives/criteria are given below. The call model should:

a. provide a high-level vendor/implementation independent abstraction of call/service processing that implies:

- a generic call model supporting all user access technologies under consideration for a given capability set, and

- uniformity of functions across multiple vendor products;

b. present an observable view of an SSF/CCF to an SCF;

c. take into account the existing base of evolvable network technology and the longer term need in its continuing evolution by providing an overall IN call and connection processing structure from which useable, coherent subsets of capabilities, as well as optional capabilities, can be defined as appropriate for a given capability set;

d. provide a framework for defining the information flows (relationships) between a SSF and an SCF, without any assumptions about the physical implementation or distribution of functions:

- this implies the need to support one or more concurrent instances of the SCF interacting with an SSF/CCF on a single call attempt (given item g. below);

e. provide a framework for defining triggering requirements;

f. provide a framework for ensuring correct sequencing of functions within an SSF/CCF; and

g. provide rules of representing and handling service logic instance interactions to support:

- multiple concurrent instances of IN service logic on a single call;

- concurrent instances of IN service logic and non-IN service logic (e.g., existing switch-based features logic) on a single call.

4.5.3.2.2 Modelling of Service Logic Processing Objectives/Criteria

The general objectives/criteria for modelling of service logic processing are given below. This modelling should:

a. provide a high-level vendor/implementation independent abstraction of service logic processing in the SCF, specialized resources in the SRF, and servic data in the SDF that implies uniformity of functions across multiple vendor products;

b. characterize the capabilities of an SRF and SDF made available to an SCF;

c. take into account the existing base of evolvable network technology and long term need in its continuing evolution by providing an overall IN service logic processing structure from which useable, coherent subsets of capabilities, as well as optional capabilities, can be defined as appropriate for a given capability set;

d. provide a framework for defining the information flows (relationship) between an SRF and an implementation or distribution of functions.

4.5.3.3 General Assumptions

4.5.3.3.1 Scope of Functional Entity Call/Service Logic Processing Models

The scope of call modelling will focus on SSF/CCF call and connection processing functions, as well as other functions required to support an IN call (i.e., specialized resource functions, status monitoring functions, data management functions, and traffic management functions). This functionality encompasses the management of IN service logic instances influencing SSF/CCF call and connection processing resources, SRF specialized resources, and SDF service data, as well as the following SSF/CCF functions: basic connection management, call management, and the management of service logic instance interactions in the SSF/CCF.

4.5.3.3.2 Relation to IN Conceptual Model

Functional entity call/service logic processing models adresses functional entities (FEs) and their relationships in the Distributed Functional Plane. Its mapping to the SIBs in the Global Functional Plane should be provided in the Stage 2 description of the SIBs.

4.5.3.3.3 Use of Functional Entity Call/Service Logic Processing Models

Functional entity call/service logic processing models provide a tool used by IN architects to model a call and to understand and describe the distribution of functions between FEs and functional entity relationships. The observable call/service logic processing functions described by functional entity modelling efforts can be used by service designers to facilitate the creation of service logic. These functions are reusable (e.g., in the form of SIBs), in that the same function can be reused for a variety of IN-supported service features. In addition, given that a robust feature interaction mechanism can be supported by the call/service logic processing models, service designers would not be constrained by previously defined IN-supported services/service features, and would also be able to reuse previously defined IN-supported service features to create new ones.

4.5.3.3.4 Other Considerations

Taking into account the call modelling objectives/criteria described in section 4.5.3.2, call modelling should also address the following needs:

a. understand and define the Distributed Functional Plane architecture;

b. define the scope of call modelling efforts for the various capability sets under study;

c. establish appropriate rules for representing and handling service logic instance interactions.

4.5.3.4 Overview of Call/Service Logic Processing Related Functional Entities

The distributed functional plane functional entities (FEs) related to call modelling and service execution are shown in Figure 32. The relationships between these functional entities are also shown. The functional entities and relationships of particular concern to call modelling include the Call Control Agent Function (CCAF), the SSF/CCF, the SRF and the relationship between the SSF/CCF and the SCF. These FEs and their relationships will be studied in terms of functional entity call/service logic processing models to identify and define functional distribution and information flows between these FEs.

The SSF and CCF are being treated together. Since there is extensive mutual interaction between the two FEs, it is anticipated that they may both be mapped into the same physical entity. As such, it is assumed that the relationship between them (i.e., the information flows) is not externally visible.

Additional subjects for study have also been identified for two cases in which multiple instances of one FE interact with another FE. These cases are shown in Figure 33 and Figure 34. In the first case, in which multiple SCFs can interact with one SSF, the SSF may have to be able to manage interactions between IN service logic instances realized in different SCFs that are simultaneously active on a single call. In the second case, in which one SCF can interact with multiple SSFs, the SCF may have to be able to manage IN call/connection processing among multiple SSFs.

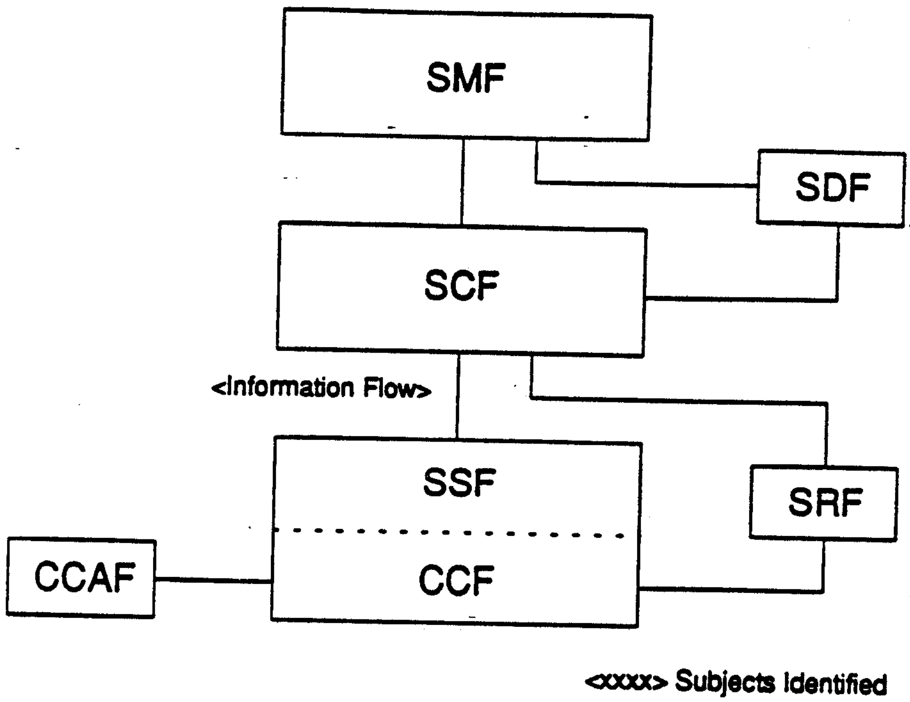


Figure 32: Overview of Call Modelling and Service Execution FEs

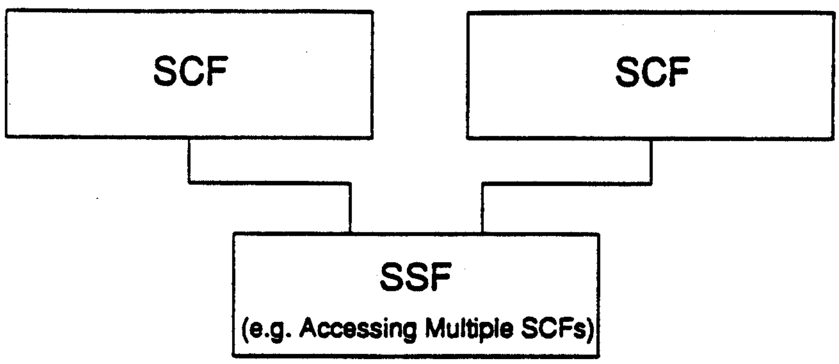


Figure 33: Multiplicity in Interaction Between SSF and SCFs

The role of the call/service logic processing functional entities in the Distributed Functional Plane architecture is as follows. Users access the call/service processing functions of the SSF/CCF via the CCAF functional entity. The CCAF receives call setup/service requests from users and passes them to the SSF/CCF for processing, independent of user access arrangements. In the course of processing these requests, the SSF/CCF may detect events (e.g., "off-hook" indicator or service feature activator) that can lead to the invocation of an instance of service logic. This instance of service logic may provide IN service features, as supported by Service Logic Processing Programs (SLPs) in an SCF, or it may provide non-IN service features, as supported by the SSF/CCF itself. Based on existing conditions and pre-specified criteria, the SSF/CCF determines if its own service logic should handle the event, or if SLPs in an SCF should handle the event. In the latter case, it reports the event to the SCF, along with the state of the call/service attempt at the time the event was detected. The SCF then invokes instances of the appropriate SLPs, and interacts with the SSF/CCF to provide instances of IN-supported service features to users. To do so, the SCF can request the SSF/CCF to perform certain call processing/service functions, and can request the

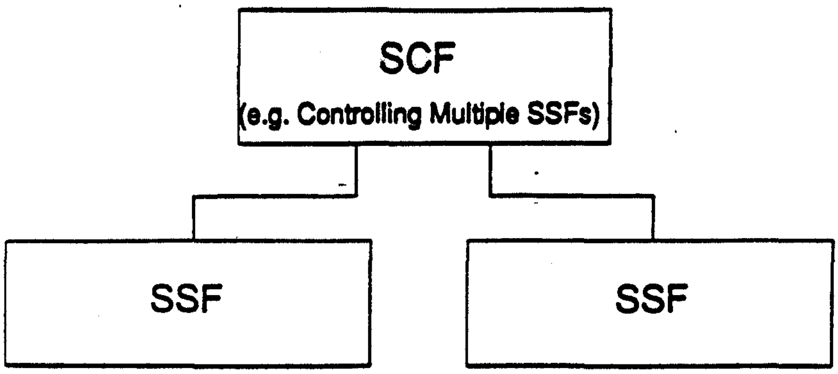


Figure 34: Multiplicity in Interaction Between SCF and SSFs

SSF/CCF to make use of resources in the SRF. In addition, it can request thhe SDF to perform related service data processing functions.

Examples of call/connection processing functions accessible to the SCF from the SSF/CCF include functions to:

a. Influence the flow of call processing (e.g., provide serial calling or generate signalling events)

b. Access and change information related to call processing (e.g., address translation)

c. Manipulate the connectivity of the call (e.g., place a party on hold, forwarding, conferencing)

d. Monitor for events related to call processing and connectivity manipulation.

Examples of specialized resource functions accessible to the SCF from the SRF include functions to:

e. Send information to users participating in a call (e.g., prompts, announcements)

f. Receive information from users participating in a call (e.g., authorization codes)

g. Modify user information (e.g., text to speech synhesis, protocol conversion)

h. Provide specialized connection resources (e.g., audio conference bridge, information distributed bridge).

Examples of service data processing functions accessible to the SCF from the SDF include functions to:

i. Access service information (e.g., subscription data parameters)

j. Update service information (e.g., sum of charging)

4.5.3.5 Call/Service Logic Processing Functional Entity Models

To address the call/service logic processing subjects identified in Sections 4.5.3.1 through 4.5.3.4 for given capability set, functional entity models are described in the context of the objectives, scope, and contraints of that capability set. To advance the understanding of these call/service logic processing subjects through functional entity modelling, IN studies have developed a number of models and modelling techniques as useful tools, independent of a given capability set. The annexes to this chapter provide some of these tools as a framework for this functional entity modelling.

While these tools have proven extremely useful in progressing IN stanardization, they do not necessarily represent an ultimate view of IN evolution. For example, the call modelling studies in Annex A currently address connection-oriented call processing; their applicability to connectionless call processing requires further study. In addition, these studies are subject to future improvement. As such, these annexes represent a current understanding that may require future enhancements.

Annex A to this chapter provides an example of an overall Basic Call State Model for existing switch processing of a basic two-party call. This model can be analyzed to identify points in basic call and connection processing when IN service logic instances are permitted to interact with basic call and connection control capabilities.

Annex B to this chapter describes an object-oriented finite state machine modelling approach for describing the dynamic aspect of funtional entities. This approach is useful in describing an external view of functional entity and its relationships with oter functional entities.

Annex C to this Recommendation describes an overall representation of call to understand how funtional entities may view and manage the call, and how interactions between multiple functional entities with respect to same call may be managed.

4.5.4 Relationship between Functional Entities

4.5.4.1 General

In support of the execution of a specific IN-supported service feature, Functional Entities (FEs) in the distributed functional plane must invoke capabilities provided by other FEs, (e.g., the SCF must invoke capabilities provided by the SSF). In order for one FE (termed the requesting FE) to invoke the capabilities provided by another FE (termed the serving FE), a relationship must be established between the two FEs concerned.

A relationship between two FEs can only be established by the client FE, though both functional entities can act as clients under different circumstances. In the former case, for example, the relationship between the SCF and the SDF to retrieve or update information can only be established at the request of the SCF. However, in the latter case of the SCF - SSF relationship, this may be established at the request of either FE (e.g., by the SCF to request the SSF to initiate a call, or by the SSF to request call processing instructions from the SCF). A relationship between two FEs can be terminated by either of the two FEs concerned.

One FE may have more than one relationship established simultaneously with other FEs.

4.5.4.2 Relationships

IN service control and service management relationships may be established between the following FEs in the distributed functional plane:

- SCF - SSF
- SCF - SRF
- SCF - SDF
- SCF - SCF
- SDF - SDF

- SMF - SSF
- SMF - CCF
- SMF - SRF
- SMF - SDF
- SMF - SCF
- SMF - SCEF
- SMF - SMAF

4.5.4.3 Information flows between FEs

Information flows between two FEs either consist of a request/response pair or of a request alone.

4.5.5 Mapping the Global Functional Plane to the Distributed Functional Plane

4.5.5.1 Mapping Requirements

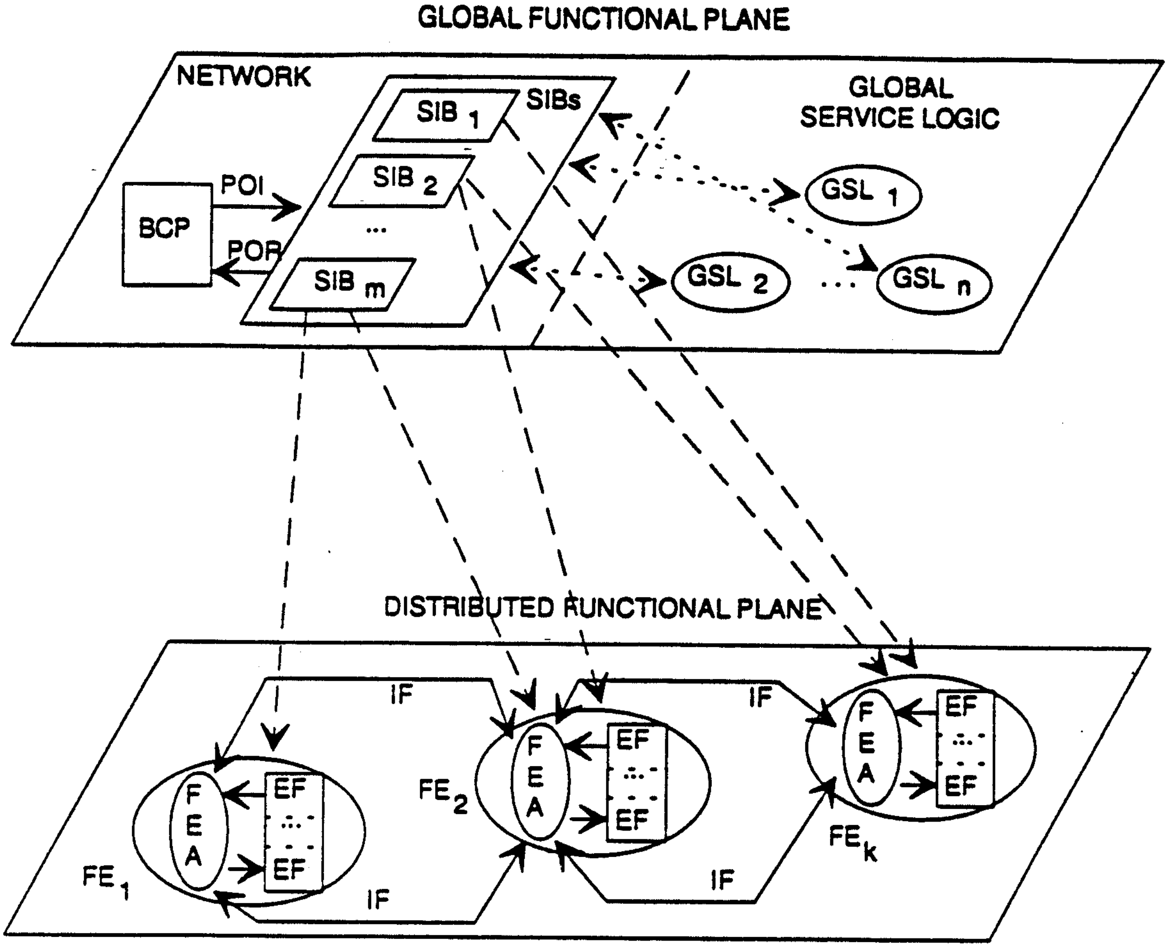
- The mapping of the Global Functional Plane to the Distributed Functional Plane must be consistent with the IN Conceptual Model;
- The mapping of the Global Functional Plane to the Distributed Functional Plane must use the Stage 2 description methodology based on the Recommendation I.130;
- The mapping of the Global Functional Plane to the Distributed Functional Plane must allow for realization of each SIB identified in the Global Functional Plane in at least one FE of the Distributed Functional Plane;
- The mapping of the Global Functional Plane to the Distributed Functional Plane must allow for realization of both Network- and Service Inter-working.
- The mapping of the Global Functional Plane to the Distributed Functional Plane must allow for accommodation of the service execution capabilities (initially), as well as for the service creation, service management, and network management capabilities, when these are introduced in the future; and
- The description of the Basic Call Process (BCP) should be completely consistent with the BCSMs that represent the call model in the Distributed Functional Plane.

4.5.5.2 Relationship to IN Conceptual Model

A set of service logic programs in the Global Functional Plane is represented by a set of distributed service logic programs located at the FEs, which are summarized in the next sub-sections.

Particularly, each SIB is realized in the Distributed Functional Plane by a sequence of specific Functional Entity Actions (FEAs) performed by the FEs. Some of these FEAs result in information flows between FEs, which is illustrated in Figure 35.

In addition, an example, that illustrates the mapping of several previously identified SIBs is presented in section 4.5.5.2.1.



1. SIB: Service Independent Building Block
2. FEA: Functional Entity Action
3. FE: Functional Entity
4. GSL: Global Service Logic
5. IF: Information Flow
6. EF: Elementary Function
7. BCP: Basic Call Process
8. POI: Point of Initiation
9. POR: Point of Return

Figure 35: Mapping of the Global Functional Plane to Distributed Functional Plane

In the DFP, the BCP can be realized in a pair of BCSMs that consists of the originating BCSM and terminating BCSM.

4.5.5.2.1 An example of mapping some selected SIBs to FEs

The following Table presents an example of mapping of SIBs to FEs. In this table, for each SIB, the involved FEs are marked with an "X".

Table: Example mapping of SIBs to Functional Entities

SIBs	Functional Entities			
	Functional Entity 1	Functional Entity 2	Functional Entity 3	Functional Entity 4
SIB 1	X	X		
SIB 2		X		X
SIB M	X	X	X	

The following Table presents an example of mapping of information flows to SIBs. In this table, for each information flow, the involved SIBs are marked with an "X".

Table: Example mapping of information Flows to SIBs

Information Flows	SIBs			
	SIB 1	SIB 2	SIB 3	SIB M
Information Flow 1	X		X	
Information Flow 2		X		X
Information Flow N	X			X

Annex A: Example Basic Call State Model (BCSM)

A.1 General

The BCSM is a high-level finite state machine description of CCF activities required to establish and maintain communication paths for users. As such, it identifies a set of basic call and connection activities in a CCF and shows how these activities are joined together to process a basic call and connection (i.e., establish and maintain a communication path for a user). The relationship between basic call and connection separation and the BCSM described in this section is for further study.

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

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

Figure 36 shows the components that have been identified to describe a BCSM, to include: Points in Connection (PICs), Detection Points (DPs), transitions, and events. PICs identify CCF activities required to complete one or more basic call/connection states of interest to IN service logic instances. DPs indicate points in basic call and connection processing at which transfer of control can occur. Transitions indicate the normal flow of basic call/connection processing from one PIC to another. Events cause transitions into and out of PICs.

Given a target description of a BCSM using these components, different subsets of PICs, DPs, transitions, and events can be identified to align with specific capability sets as they are defined. In addition, though CCAF functionality is not explicitly modelled in the BCSM, a mapping is required between access signalling events and BCSM events, for each access arrangement supported by a given capability set. Since the BCSM is generic, it may describe events that do not apply to certain access arrangements. It is important to understand how each access arrangement applies to the BCSM, and may be desireable to show in separate representations the aspects of the BCSM that apply to each arrangement.

A.2 Example BCSM Description

An example of an overall BCSM for existing switching processing of basic two-party calls described in this section and is illustrated in Figure 36 and Figure 37. The example does not imply the aspects of the BCSM that are visible to IN service logic instances, or the nature of the information flows between the SSF/CCF and SCF. This example may not represent the ultimate evolution of the BCSM for CS-N, but does provide a starting point for identifying PICs and DPs for a given capability set, based on service requirements.

In the following descriptions, the PICs are related at a high level to Q.931 ISDN Call States. This is not intended to be a detailed formal definition of the relation between the PICs and Q.931 ISDN Call States, but is intended as a point of reference to use in understanding the PICs. In particular, there are a number of possible ways in which the Q.931 Call States may be traversed in certain situations which are not considered below.

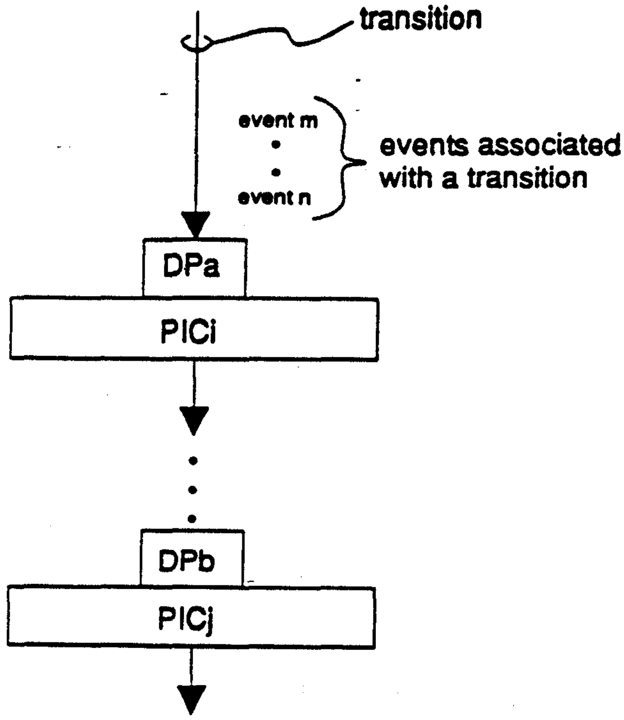


Figure 36: BCSM Components

In order to maintain uniqueness of DP names between the originating and terminating half BCSMs, 'O' and 'T' is prefixed to certain originating and terminating DP names, respectively.

Certain PICs correspond to switch-based service feature functionality, and thus are not ubiquitous among all switching systems. They are denoted "optional" to reflect the current understanding. For future Capability Sets, it may be desirable to incorporate this functionality fully into the BCSM, depending on the benchmark set of services for given Capability Sets.

For ease of reference, the DPs associated with the transition implied by each entry and exit event for each PIC are listed along with the PIC descriptions. When in PIC processing an exit event is detected is for further study.

a. Example Originating BCSM

The originating half of the BCSM corresponds to that portion of the BCSM associated with the originating party. It is shown in Figure 37. The description for each of the PICs in the originating half of the BCSM are described below:

1. O_Null

Entry Event: Disconnect and clearing of a previous call. (DPs 20 - O_Disconnect_Complete or 21 - O_Abandon & O_Calling_Party_Disconnect)

Function: Interface (line/trunk) is idled (no call exists, no call reference exists, etc.) Supervision is being provided.

Exit Event: Indication of desire to place outgoing call received from originating party (e.g., offhook, Q.931 Setup message, ISDN-UP IAM message) (DP 1 - Origination_Attempt)

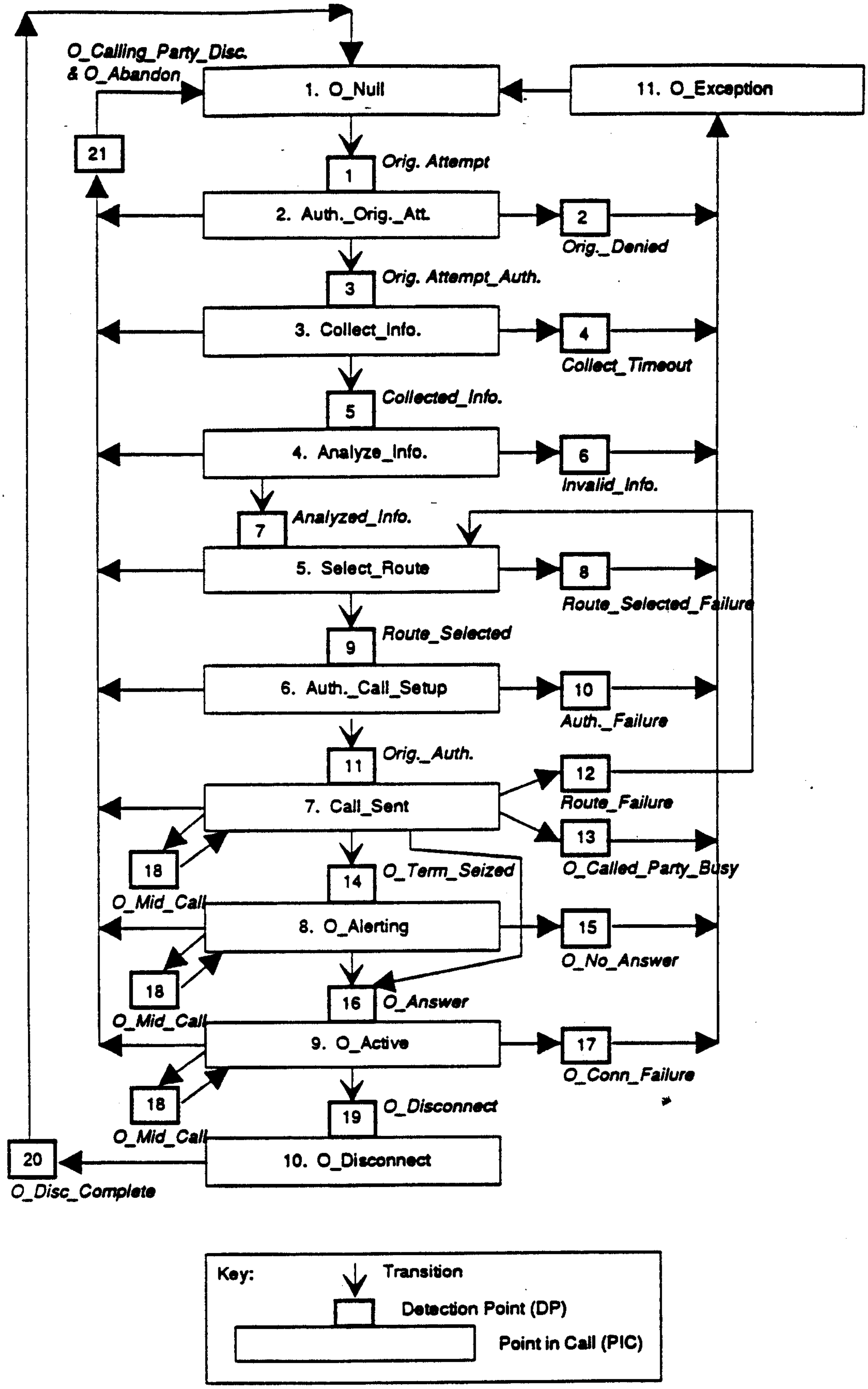


Figure 37: Example Orginating BCSM

Corresponding Q.931 Call State: 0. Null

2. Authorize_Origination_Attempt (optional)

Entry Event: Indication of desire to place outgoing call received from originating party (DP 1 - Origination_Attempt)

Function: Authority/ability of originating party to place outgoing call with given properties (e.g., bearer capability, line restrictions) is being verified. The types of authorization to be performed may vary for different types of originating resources (e.g., for lines vs. trunks).

Exit Events:

- Authority/ability to place outgoing call denied (DP 2 - Origination_Denied)

- Authority/ability to place outgoing call verified (DP 3 - Origination_Attempt_Authorized)

- Originating party abandons call. (DP 21 - O_Abandon & O_Calling_Party_Disconnect)

Comment: Originating party abandon may not be detected until the end of processing for this PIC, since this PIC may be processed in an atomic manner (i.e., processing cannot be interrupted) for some implementations.

Corresponding Q.931 Call State: Not Applicable

3. Collect_Information

Entry Event: Authority/ability to place outgoing call verified (DP 3 - Origination_Attempt_Authorized)

Function: Initial information package/dialing string (e.g., service codes, prefixes, dialed address digits) being collected from originating party. Information being interpreted according to dialing plan to determine end of collection. No further action may be required if an en bloc signaling method is in use (e.g., an ISDN user using en bloc signaling, an incoming SS7 trunk).

Exit Events:

- Information collection error has occurred (e.g., invalid dial string format, digit collection timeout) (DP 4 - Collect_Timeout)

- Availability of complete initial information package/dialing string from originating party. (This event may have already occurred in the case of en bloc signaling, in which case the waiting duration in this PIC is zero.) (DP 5 - Collected_Info)

- Originating party abandons calls. (DP 21 - O_Abandon & O_Calling_Party_Disconnect)

Comment: Some digit analysis is required to determine the end of dialing. However, it is assumed that this analysis may be modelled as separable from the rest of digit analysis, which occurs in PIC 4, Analyze_Information. There is no intention to specify an implementation. However, a switch should externally present the separable view described for closed numbering plans[1].

Corresponding Q.931 Call State: 1. Call Initiated and (optionally) 2. Overlap Sending

4. Analyze_Information

Entry Event: Availability of complete initial information package/dialing string from originating party. (DP 5 - Collected_Info)

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

Exit Events:

- Unable to interpret and translate dial string in the dialing plan (e.g., invalid dial string) (DP 6 - Invalid_Info)
- Availability of routing address and call type. (DP 7 - Analyzed_Info)
- Originating party abandons call. (DP 21 - O_Abandon & O_Calling_Party_Disconnect)

Comments:

- Note that routing address does not necessarily mean that the final physical route has been determined (e.g., route list has not been searched, hunt groups have not yet been searched, DN has not yet been translated to physical port address), though this may be the case (e.g., when routing to a specific private facility).
- Originating party abandon may not be detected until the end of processing for this PIC, since this PIC may be processed in an atomic manner (i.e., processing cannot be interrupted) for some implementations.

Corresponding Q.931 Call State: Not Applicable

5. Select_Route

Entry Events:

- Availability of routing address and call type. (DP 7 - Analyzed_Info)
- Unable to complete call using specified route (e.g., congestion) (DP 12 - Route_Failure)

Function: Routing address and call type being interpreted. The next route is being selected. This may involve sequentially searching a route list, translating a DN into physical port address, etc. The individual destination resource out of a resource group (e.g., a multi-line hunt group, a trunk group) is not selected. In some cases (e.g., a normal POTS line), a single resource (not a group) is selected.

Exit Events:

---

1) This seperable view is provided by supporting distinct DPs for DP 5 (Collected_Info) and DP 7 (Analyzed_Info), and by populating information flows accordingly for corresponding TDP and EDP information flows to the SCF.

- Unable to select a route (e.g., unable to determine a correct route, no more routes on route list) (DP 8 - Route_Select_Failure)

- Terminating resource (group) to which call should be routed has been identified (DP 9 - Route_Selected)

- Originating party abandons call. (DP 21 - O_Abandon & O_Calling_Party_Disconnect)

Corresponding Q.931 Call State: Not Applicable

6. Authorize_Call_Setup (optional)

Entry Event: Terminating resource (group) to which call should be routed has been identified (DP 9 - Route_Selected)

Function: Authority of originating party to place this particular call being verified (e.g., checking business group restrictions, toll restrictions, route restrictions). The types of authorization checks to be performed may depend upon the type of originating resource (e.g., line vs. trunk).

Exit Events:

- Authority of originating party to place this call is denied (e.g., business group restriction mismatch, toll restricted calling line) (DP 10 - Authorization_Failure)

- Authority of originating party to place this call verified. (DP 11 - Origination_Authorized)

- Originating party abandons call. (DP 21 - O_Abandon & O_Calling_Party_Disconnect)

Corresponding Q.931 Call State: Not Applicable

7. Call_Sent

Entry Event: Authority of originating party to place call verified (DP 11 - Origination_Authorized)

Function: Call is being processed by the terminating half BCSM. The originating half BCSM is awaiting some indication that the call has been presented to the called party.

Exit Events:

- Indication from terminating half BCSM that the call cannot be presented to the terminating party (e.g., network congestion) (DP 12 - Route_Failure)

- Indication from terminating half BCSM that the terminating party is busy (DP 13 - O_Called_Party_Busy)

- Indication from terminating half BCSM that the terminating party is being alerted (DP 14 - O_Term_Seized)

- Indication from terminating half BCSM that the call is accepted and answered by terminating party (e.g., terminating party goes offhook, Q.931 Connect message received, ISDN-UP Answer message received) (DP 16 - O_Answer)

- A service/service feature request is received from the originating party (e.g., hook flash, ISDN feature activator, Q.931 HOLD or RETrieve message). (DP 18 - O_Mid_Call)

- Originating party abandons call. (DP 21 - O_Abandon & O_Calling_Party_Disconnect)

Corresponding Q.931 Call State: Not Applicable

8. O_Alerting

Entry Event: Indication from terminating half BCSM that the terminating party is being alerted of incoming call (DP 14 - O_Term_Seized)

Function: Continued processing of call setup (e.g., power ringing, audible ring indication) is taking place. Waiting for indication from terminating half BCSM that the call has been answered by terminating party.

Exit Events:

- Indication from terminating half BCSM that the terminating party does not answer within a specified time period (DP 15 - O_No_Answer)
- Indication from terminating half BCSM that the call is accepted and answered by terminating party (e.g., terminating party goes offhook, Q.931 Connect message received, ISDN-UP Answer message received) (DP 16 - O_Answer)
- A service/service feature request is received from the originating party (e.g., hook flash, ISDN feature activator, Q.931 HOLD or RETrieve message). (DP 18 - O_Mid_Call)
- Originating party abandons call. (DP 21 - O_Abandon & O_Calling_Party_Disconnect)

Comment: This PIC is not applicable when terminating to a non-SS7 trunk group. For terminations to SS7 trunk groups, this PIC is entered upon the receipt of an Address Complete (ACM) message.

Corresponding Q.931 Call State: 4. Call Delivered

9. O_Active

Entry Event: Indication from terminating half BCSM that the call is accepted and answered by terminating party. (DP 16 - O_Answer)

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

Exit Events:

- A connection failure occurs (DP 17 - O_Connection_Failure)
- A service/service feature request is received from the originating party (e.g., DTMF hook flash, ISDN feature activator, Q.931 HOLD or RETrieve message). (DP 18 - O_Mid_Call)
- A disconnect indication (e.g., onhook, Q.931 Disconnect message, SS7 Release message) is received from the originating party. (DP 21 - O_Abandon & O_Calling_Party_Disconnect)
- A disconnect indication (e.g., onhook, Q.931 Disconnect message, SS7 Release message) is received from the terminating party via the terminating half BCSM. (DP 19 - O_Disconnect)

Comment: A terminating party may disconnect then reconnect before the expiration of disconnect timing. In this case, the call is considered to remain in the O_Active PIC.

Corresponding Q.931 Call State: 10. Active

10. O_Disconnect

Entry Event: A disconnect indication is received from the originating party, or received from the terminating party via the terminating half BCSM. (DP 19 - O_Disconnect)

Function: Disconnect treatment is being applied[1].

Exit Event: Completion of disconnection of call (e.g., expiration of disconnect timing, resources idled) (DP 20 - O_Disconnect_Complete)

Comment: Disconnect indications and treatment are asymmetrical in the way disconnect timing is applied. Disconnect treatment and timing is different for ISDN and POTS calls.

Corresponding Q.931 Call States: 11. Disconnect Request, 12. Disconnect Indication and 19. Release Request.

11. O_Exception

Entry Event: An exception condition is encountered (DPs 2, 4, 6, 8, 10, 13, 15, and 17 as described above)

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

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

- If an SCF previously requested that call parameters be provided at the end of the call (e.g., for logging purposes), these should be included in the Error information flow.

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

Exit Event: Default handling of the exception condition by SSF/CCF completed (Transition to O_Null).

b. Example Terminating BCSM

The terminating half of the BCSM corresponds to that portion of the BCSM associated with the terminating party. It is shown in Figure 38. The description for each of the PICs in the terminating half of the BCSM are described below:

---

1) If the terminating party disconnected (i.e., an indication of terminating party disconnect is received from the terminating half BCSM), then disconnect timing may be applied by the terminating half BCSM (if the terminating party is not an ISDN station).

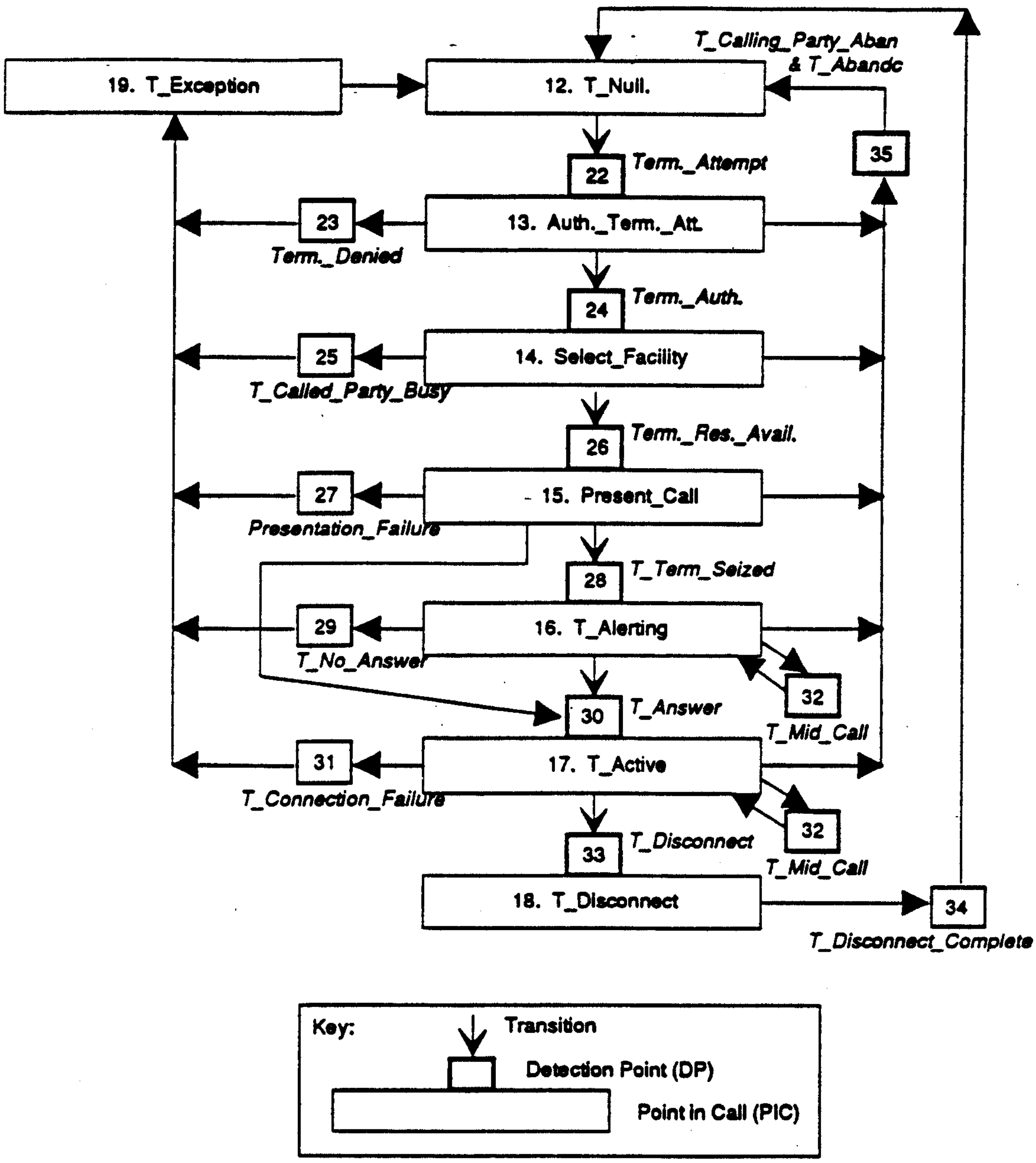


Figure 38: Example Terminating BCSM

12. T_Null

Entry Event: Disconnect and clearing of a previous call. (DPs 34 - T_Disconnect_Complete or 35 - T_Abandon & T_Calling_Party_Disconnect)

Function: Interface (line/trunk) is idled (no call exists, no call reference exists, etc.) Supervision is being provided.

Exit Event: Indication of incoming call received from originating half BCSM (DP 22 - Termination_Attempt)

Corresponding Q.931 Call State: 0. Null

13. Authorize_Termination_Attempt (optional)

    Entry Event: Indication of incoming call received from originating half BCSM (DP 22 - Termination_Attempt)

    Function: Authority to route this call to terminating party is being verified (e.g., business group restrictions, restricted incoming access to line, bearer capability compatibility). (This PIC may not be applicable for terminations to trunks.)

    Exit Events:

    - Authority to route call to specified terminating resource (or group) denied. (DP 23 - Termination_Denied)
    - Authority to route call to a specified terminating resource (or group) verified. (DP 24 - Termination_Authorized)
    - Indication of originating party abandon received from originating half BCSM. (DP 35 - T_Abandon & T_Calling_Party_Disconnect)

    Comment: Originating party abandon may not be detected until the end of processing for this PIC, since this PIC may be processed in an atomic manner (i.e., processing cannot be interrupted) for some implementations.

    Corresponding Q.931 Call State: Not Applicable

14. Select_Facility

    Entry Event: Authority to route call to specified terminating resource group verified. (DP 24 - Termination_Authorized)

    Function: A particular available resource in the specified resource group is being selected. It is possible that all resources in the group could be busy. A single resource is treated as a group of size 1.

    Exit Events:

    - All resources in group busy or called party busy. (DP 25 - T_Called_Party_Busy)
    - Available terminating resource in resource group identified. (DP 26 - Terminating_Resource_Available)
    - Indication of originating party abandon received from originating half BCSM. (DP 35 - T_Abandon & T_Calling_Party_Disconnect)

    Corresponding Q.931 Call State: Not Applicable

15. Present_Call

    Entry Event: Available terminating resource identified. (DP 26 - Terminating_Resource_Available)

Function: Terminating resource informed of incoming call (e.g., line seizure, Q.931 Setup message, ISDN-UP IAM message). In the case of an analog line, power ring is in the process of being applied.

Exit Events:

- Cannot present call (e.g., ISDN user determined busy, ISDN-UP Release message with busy cause) (DP 27 - Presentation_Failure)

- Terminating party is being alerted (e.g., power ring being applied, Q.931 Alerting message, ISDN-UP ACM message). In the case of an analog line, this event should occur almost instantly after entering this PIC. (DP 28 - T_Termination_Seized)

- Call is accepted and answered by terminating party (e.g., terminating party goes offhook, Q.931 Connect message received, ISDN-UP Answer message received) (DP 30 - T_Answer)

- Indication of originating party abandon. (DP 35 - T_Abandon & T_Calling_Party_Disconnect)Corresponding Q.931 Call State: 6. Call Present

16. T_Alerting

Entry Event: Terminating party is being alerted of incoming call (DP 28 - T_Termination_Seized)

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

Exit Events:

- Terminating party does not answer within a specified duration. (DP 29 - T_No_Answer)

- Call is accepted and answered by terminating party (e.g., terminating party goes offhook, Q.931 Connect message received, ISDN-UP Answer message received) (DP 30 - T_Answer)

- A service/service feature request is received from the terminating party (e.g., hook flash, ISDN feature activator, Q.931 HOLD or RETrieve message). (DP 32 - T_Mid_Call)

- Indication of originating party abandon received from originating half BCSM. (DP 35 - T_Abandon & T_Calling_Party_Disconnect)

Comment: This PIC is not applicable when terminating to a non-SS7 trunk group. For terminations to SS7 trunk groups, this PIC is entered upon the receipt of an Address Complete (ACM) message.

Corresponding Q.931 Call States: 7. Call Received and 8. Connect Request

17. T_Active

Entry Events: An indication is sent to the originating half BCSM that the terminating party has accepted and answered the call (e.g., terminating party goes offhook, Q.931 Connect message received, ISDN-UP Answer message received) (DP 30 - T_Answer)

Function: Connection established between originating and terminating party. Call supervision is being provided.

Exit Events:

- A connection failure occurs. (DP 31 - T_Connection_Failure)
- A service/service feature request is received from the terminating party (e.g., hook flash, ISDN feature activator, Q.931 HOLD or RETrieve message). (DP 32 - T_Mid_Call)
- A disconnect indication (e.g., onhook, Q.931 Disconnect message, SS7 Release message) is received from the terminating party (DP 33 - T_Disconnect).
- A disconnect indication (e.g., onhook, Q.931 Disconnect message, SS7 Release message) is or received from the originating party via the originating half BCSM. (DP 35 - T_Abandon & T_Calling_Party_Disconnect)

Comment: A terminating party may disconnect then reconnect before the expiration of disconnect timing. In this case, the call is considered to remain in the T_Active PIC.

Corresponding Q.931 Call State: 10. Active

18. T_Disconnect

Entry Event: A disconnect indication is received from the terminating party, or received from the originating party via the originating half BCSM. (DP 33 - T_Disconnect)

Function: Disconnect treatment being applied. This includes disconnect timing for non-ISDN calls.

Exit Events:

- Completion of disconnection of call. (DP 34 - T_Disconnect_Complete)

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

Corresponding Q.931 Call States: 11. Disconnect Request, 12. Disconnect Indication, and 19. Release Request

19. T_Exception

Entry Event: An exception condition is encountered (DPs 23, 25, 27, 29, and 31 as described above)

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

- If any relationships exist between the SSF and SCF(s), send an Error information flow to the SCF(s) closing the relationships and indicating that any outstanding call handling instructions will not run to completion .
- If an SCF previously requested that call parameters be provided at the end of the call (e.g., for logging purposes), these should be included in the Error information flow.

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

Exit Event: Default handling of the exception condition by SSF/CCF completed (Transition to T_Null).

Annex B: Object-oriented Finite State Machine Modelling

A Finite State Machine (FSM) model is an operational (dynamic) model of a system that is described by the finite set of states the system can be in and the finite set of transitions possible from one state to another state. Specific operations or activities being performed by the system at a given point in time are associated with specific states. Specific inputs to or outputs from a given state (referred to as events) are associated with each transition from one state to another state. For a given event, the transition from one state to another will always be the same (i.e., the FSM model is deterministic).

To describe a system in terms of an FSM model, it is first necessary to characterize the state of the system. Object-oriented techniques provide a means to do so by first characterizing the system as a set of one or more objects that represent the system properties of interest. These abstract objects are each described in terms of the unique characteristics they represent (referred to as attributes) and the actions that can manipulate those objects (referred to as functions). As a result, the objects are modular, autonomous entities that can be freely combined and reused. In addition, their description can be at an appropriate level of abstraction to hide the technical complexity, physical details, and evolution of the underlying technology of the system.

Given the set of objects that characterize the system (e.g., legs and connection points), the state of the system and its operation can be described. For a given instance of an object, its attributes have specific values which can be constant (i.e., static) or variable (i.e., dynamic). The state of an object instance is defined by the values of its attributes at a given point in time. The state of a system is then defined by the set of states of its objects. The operation of the system can then be described in terms of an FSM model, given that transitions are defined as changes in the values of object attributes, which changes the state of the system. Transitions occur when events cause the system to perform functions that change the values of object attributes.

The benefits of the object-oriented FSM modelling approach are as follows:

a. it provides a means of completely characterizing a system in terms of a set of modular objects, and its operation in terms of a finite set of well-defined states and transitions;

b. it allows the system to be described at an appropriate level of abstraction to model the objects, attributes, functions, and events of interest;

c. it provides an external (observable) view of a system in terms of the functions it performs in response to events, a view which hides the implementation details of how the system performs these functions;

d. it can be formally specified.

An object-oriented FSM modelling approach is in general appropriate whenever an external view of a system is desired (i.e., the view of an SSF from an SCF). In particular, it is useful in describing a relationship or an information flow between systems, and is often used to describe protocol machines at a system interface (where protocol messages serve as events). The benefits of this FSM modelling approach serve to meet the general call modelling objectives/criteria described in Section 4.5.3.2. In particular, it helps to:

e. provide a high-level, vendor/implementation independent abstraction of call/service processing functions;

f. present an observable view of an SSF to an SCF;

g. provide a framework for defining the information flows between the SSF and SCF;

h. provides a framework for ensuring the correct sequencing of functions within an SSF.

The following methodology can be applied to define IN object types in keeping with an object-oriented FSM modelling approach:

Step 0 - Identification of IN Object Type

IN object types are identified based on the following rules:

a. IN object types should be identified to represent the physical resources of interest (i.e., physical resources accessible via an interface);

b. separate IN object types should be identified for physical resources whose behaviour is represented by different sets of IN object states and/or different object state transitions;

Step 1 - Definition of IN Object Types

IN object types are defined in terms of functionality and physical resource mapping. The functionality and physical resource mapping should be expressed in terms of both the attributes that characterize the resource and the functions that manipulate the resource.

Step 2 - State Description of IN Object Types

The state of an IN object type is defined in terms of the values of its attributes. The values of the attributes at a given point in time define the state of an IN object. The allowed range and combination of attribute values should be described to define the set of possible states for an IN object type.

Step 3 - IN Object Type Functional Message Definition

An IN object type functional message invokes a single IN object function or a set of IN object functions. IN object functions can cause transitions from one IN object state to another by changing the values of IN object attributes. Functional messages should be defined for an IN object type to invoke each IN object function or set of functions of interest.

Step 4 - IN Object Type State Transitions

IN object type state transitions are caused by IN object functions in response to IN object functional messages. SDL diagrams should be developed to describe the state transitions of IN object types in response to functional messages.

Annex C: Call Segment Model

C.1 Call Segment Model Components

An overall representation of a call is useful to understand how the SSF/CCF views and manages a call. A Call Segment Model (CSM) provides such a representation. The components of the CSM that have been identified include access segments, basic call segments, feature segments and links. Figure 39 illustrates these components. Access segments represent external access points into or out of an entity containing SSF/CCF functionality (e.g., line or trunk access arrangements). Basic call segments represent the relationship between access arrangements(as supported by switching and transmission resources and described by the BCSM). As such, there are basic call segments between access segments. Since functional separation is maintained between originating and terminating basic call segments, these segments provide a means to isolate one end user's call/service processing from another (to maintain single-ended control), and to manage signals propagating from one access segment to another. A feature segment corresponds to an instance of service logic invoked on behalf of a particular end user. It is inserted between the access segment and basic call segment associated with the end user. Links represent the signalling and transmission paths between segments.

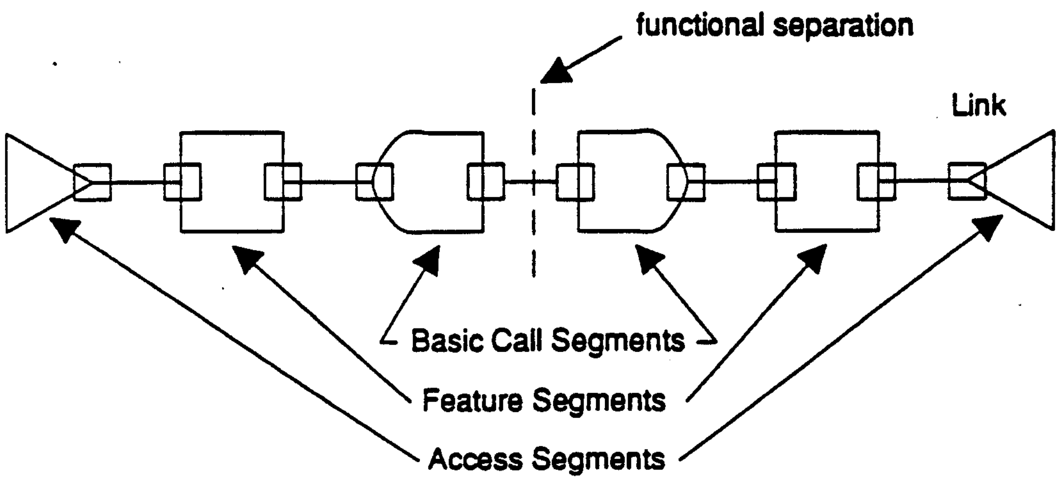


Figure 39: CSM Components

In the process of establishing and maintaining a call and invoking instances of service logic relative to the call, segments are chained together, inserted, and deleted, as appropriate.

C.2 Local/Global SCF View of Call/Connection Processing

It is recognized that the SCF can have a local view of call/connection processing via an interaction with a single SSF, but can also have a global or network view of call/connection processing via interactions with multiple SSFs. The CSM can be used to illustrate these views, as shown in Figure 40. In the local view, a service logic processing program instance in an SCF is shown interacting with a single feature segment on a call. In the global view, a service logic processing program instance in an SCF (or pair of communicating service logic processing program instances in one or more SCFs) is shown interacting with multiple feature segments on a call.

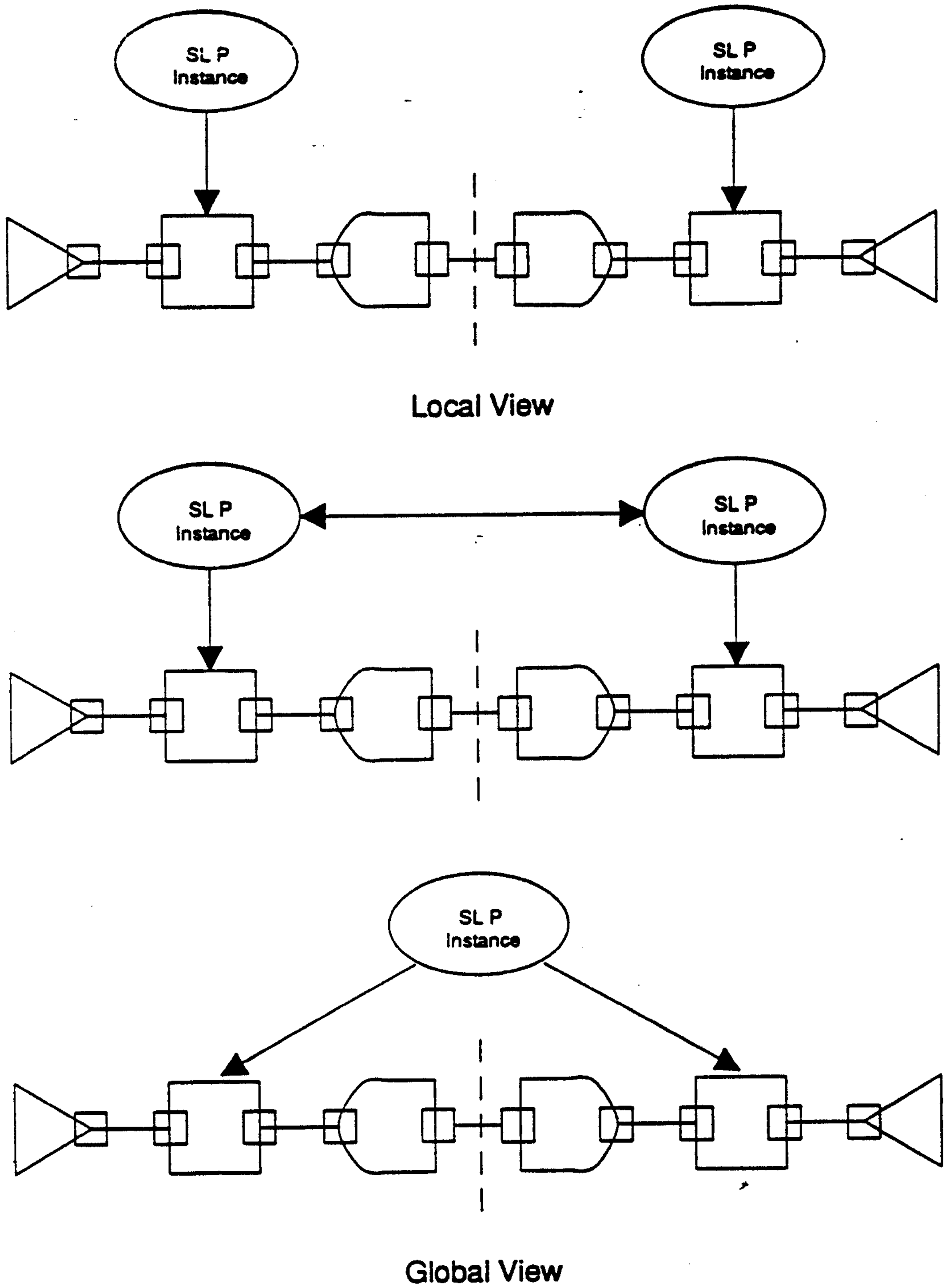


Figure 40: Local/Global SCF View of Call/Connection Processing

4.6 Physical Plane Architecture

4.6.1 General

This section describes the physical plane of the general IN architecture. IN physical plane information specific to CS-1 is contained in seperate ETS.

The physical plane of the IN Conceptual Model identifies the different physical entities and the interfaces between these entities.

The ~~physical plane~~ architecture should be consistent with the IN Conceptual Model. The IN Conceptual Model is a tool that can be used to design the IN architecture to meet the following main objectives:

- service implementation independence;

- network implementation;

- vendor and technology independence.

The I.130 Stage 3 service description methodology may be used (which includes the functional specification of the node and detailed description of the protocol between the nodes) in developing the physical plane architecture.

4.6.2 Requirements and assumptions

4.6.2.1 Requirements

The key requirements of the physical plane architecture are:

- the functional entities in the distributed functional plane can be mapped onto the physical entities;

- one or more functional entities may be mapped onto the same physical entity;

- one functional entity cannot be split between two physical entities (i.e., the functional entity is mapped entirely within a single physical entity);

- duplicate instances of functional entity can be mapped to different physical entities, though not to the same physical entity;

- physical entities can be grouped to form a physical architecture;

- the physical entities may offer standard interfaces;

- vendors must be able to develop physical entities based on the mapping of funtional entities and the standard interfaces;

- vendors must be able to support mature technologies and new technologies as they become available.

4.6.2.2 Assumptions

The following assumptions are made for the development of the physical plane architecture:

- the IN Conceptual Model is used as a tool to develop the IN physical architecture;

- existing and new technologies can be used to develop the physical entities;

- the specification of functional entities in the distributed functional plane and standard interfaces in the physical plane will make the network vendor independent and service independent;

- sufficient number of interfaces will be indentified for support of services, service creation and OAM functions.

4.6.3 Physical Entities (PEs)

This section describes a selection of PEs to support the general IN. That selection is not intended to preclude or disallow the application of any other IN PEs to support the general IN.

a) Service Switching Point (SSP)

In addition to providing users with acess to the network (if the SSP is a local exchange) and performing any necessary switching functionality, the SSP allows full access to the set of IN capabilities. The SSP contains Detection Capability to detect requests for IN services. It also contains capabilities to communicate with other PE(s) containing a Service Control Function (SCF), such as a Service Control Point (SCP), and to respond to instructions from the other PE. Functionally, an SSP contains a Call Control Function (CCF), a Service Switching Function (SSF), and, if the SSP is a local exchange, a Call Control Agent Function (CCAF): It also may optionally contain a Service Control Function (SCF), and/or a Specialized Function (SRF), and/or a Service Data Function (SDF).

b) Service Control Point (SCP)

The SCP contains the Service Logic Programs (SLPs) that are used to provide IN services, and may optionally contain customer data. Multiple SCPs may contain the same SLPs and data to improve service reliability and to facilitate load sharing between SCPs. Functionally, an SCP contains a Service Control Function (SCF), and may optionally contain a Service Data Function (SDF). The SCP can access data in an SDP either directly or through a signalling network. The SDP may be in the same network as the SCP, or in another network. The SCP can be connected to SSPs, and optionally to IPs, through the signalling network. The SCP can also be connected to an IP via an SSP relay function.

c) Service Data Point (SDP)

The SDP contains data used by SLPs to provide individualized services. Functionally, an SDP contains a Service Data Function. It can be accessed directly by an SCP and/or SMP, or through the signalling network. It can also access other SDPs in its own or other networks.

d) Intelligent Peripheral (IP)

The IP provides special resources for customization of services, and supports flexible information interactions between a user and the network. Optionally, the switching fabric used to connect users to these resources may be accessible to external SLPs. Examples of possible special resources are (this list is not meant to be exhaustive):

- customized and concatenated voice announcements

- synthesized voice/speech recognition devices

- DTMF digit collection

- audio conference bridge
- information distribution bridge
- tone generator
- text to speech synthesis
- protocol converters

The IP contains the Special Resource Function (SRF), and optionally a Service Switching Function/Call Control Function (SSF/CCF). This optional SSF/CCF is used to provide external access to the connections to the resources within the IP. The IP connects to one or more SSPs, and/or to the signalling network.

An SCP can request an SSP to connect a user to a resource located in an IP that is connected to the SSP from which the service request is detected. An SCP can also request the SSP to connect a user to a resource located in an IP that is connected to another SSP.

e) Adjunct (AD)

The Adjunct PE is functionally equivalent to an SCP (i.e., it contains the same FEs) but it is directly connected to an SSP. Communication between an Adjunct and an SSP is supported by a high speed interface. This arrangement may result in differing performance characteristics for an Adjunct and an SCP. The application layer message are identical in content to those carried by the signalling network to an SCP.

An Adjunct may be connected to more than one SSP and an SSP may be connected to more than one Adjunct.

f) Service Node (SN)

The SN can control IN services and engage in flexible information interactions with users. The SN communicates directly with one or more SSPs, each with a point-to-point signalling and transport connection. Functionally, the SN contains an SCF, SDF, and SRF. This SSF/CCF is closely coupled to the SCF within the SN, and is not accessible by external SCFs.

In a manner similar to an Adjunct, the SCF in an SN receives messages from the SSP, executes SLPs, and sends messages to the SSP. SLPs in an SN may be developed by the same service creation environment used to develop SLPs for SCPs and Adjuncts. The SRF in an SN enables the SN to interact with users in a manner similar to an IP. An SCF can request the SSF to connect a user to a resource located in an SN that is connected to the SSP from which the service request is detected. An SCF can also request the SSP to connect a user to a resource located in an SN that is connected to another SSP.

g) Service Switching and Control Point (SSCP)

The SSCP is a combined SCP and SSP in a single node. Functionally, it contains an SCF, SDF, CCAF, CCF, and SSF. The connection between the SCF/SDF functions and the CCAF/CCF/SSF functions is proprietary and closely-coupled, but it provides the same service capability as an SSP and SCP separately.

This node may also contain SRF functionality, i.e. SRF as an optional functionality.

The interfaces between the SSCP and other PEs are the same as the interfaces between the SSP and other PEs, and therefore will not be explicititly stated.

h) Service Management Point (SMP)

The SMP performs service management control, service provision control, and service deployment control. Examples of functions it can perform are database administration, network surveillance and testing, network traffic management, and network data collection. Functionally, the SMP contains the Service Management Function (SMF) and, optionally, the Service Management Access Function (SMAF) and the Service Creation Environment Function (SCEF). The SMP can access all other PEs.

i) Service Creation Environment Point (SCEP)

The SCEP is used to define, develop, and test an IN service, and to input it into the SMP. Functionally, it contains the Service Creation Environment Function (SCEP). The SCEP interacts directly with the SMP.

j) Service Creation Environment Point (SCEP)

The SMAP provides some selected users, such as service managers or some customers, with access to the SMP. One possible use of the SMAP is to provide one single point of access for a given user to several SMPs. The SMAP functionally contains a Service Management Access Function (SMAF). The SMAP interacts directly with thhe SMP.

4.6.4 Mapping the Distributed Functional Plane to the Physical Plane

4.6.4.1 Mapping of FEs to PEs

This section gives a mapping of Functional Entities (FEs) into Physical Entities (PEs), and describes the reference points between the PEs. In so doing, an appropriate distribution of functionality is identified, and functional interfaces suitable for standardization are highlighted, The PEs described in this section are for illustrative purpose only, and do not imply the only possible mapping of functionality.

This section describes a flexible physical architecture made up of several PEs. Each PE contains one or more FEs, which define its IN functionality. PEs included in the physical architecture shown in Figure 41 are SSP, SCP, SDP, IP, AD, SN, SMP, and SCEP.

Typical scenarios of FE mapping to PE are shown in the Table.

PEs:	FEs:						
	SCF	SSF/CCF	SDF	SRF	SMF	SCEF	SMAF
SSP	o	c	o	o	-	-	-
SCP	c	-	o	-	-	-	-
SDP	-	-	c	-	-	-	-
IP	-	o	-	c	-	-	-
AD	c	-	c	-	-	-	-
SN	c	-	c	c	-	-	-
SSCP	c	c	c	o			
SMP	-	-	-	-	c	-	c

SCEP	-	-	-	-	-	c	-
SMAP	-	-	-	-	-	-	c

c: core
o: optional
-: not allowed

Table: Typical scenarios of FE to PE mapping.

There is no intention that the Table should disallow other combination of FEs that would result in a PE not shown in the Table.

The above mapping are shown in Figure 41. Each PE has certain FE(s) mapped into it. As indicated in the legend of the figure, transport paths, signalling paths (that carry application layer messages), and management, provisioning and control paths are differentiated.

4.6.4.2 Selection of underlying protocol platforms

Intelligent Network Recommendations largely focus on the definition anf specification of application layer protocol to effect the scope of IN functional relationship. For any given capability set, the appropriate underlying protocol platforms are selected from those that are currently available or planned, selecting and adapting to those whose capabilities best meet the IN signalling needs.

4.6.5 User interface

A user is an entity external to thhe IN that uses IN capabilities. IN users may employ new or existing access interfaces to invoke various IN service capabilities.

It is important to ensure that IN continues to support existing services and capabilities. In addition, restrictions imposed by individual interface technologies must be considered when deploying IN services.

FEs:

CCF Call Control Function
CCAF Call Control Agent Function
SCF Service Control Function
SDF Service Data Function
SRF Special Resource Function
SSF Service Switching Function
SMF Service Management Function
SCEF Service Creation Environment Function
SMAF Service Management Access Function

PEs:

SSP Service Switching Point
SCP Service Control Point
SDP Service Data Point
IP Intelligent Peripheral
SMP Service Management Point
SCEP Service Creation Environment Point
AD Adjunct
SN Services Node
SSCP Service Switching and Control Point
SMAP Service Management Access Point

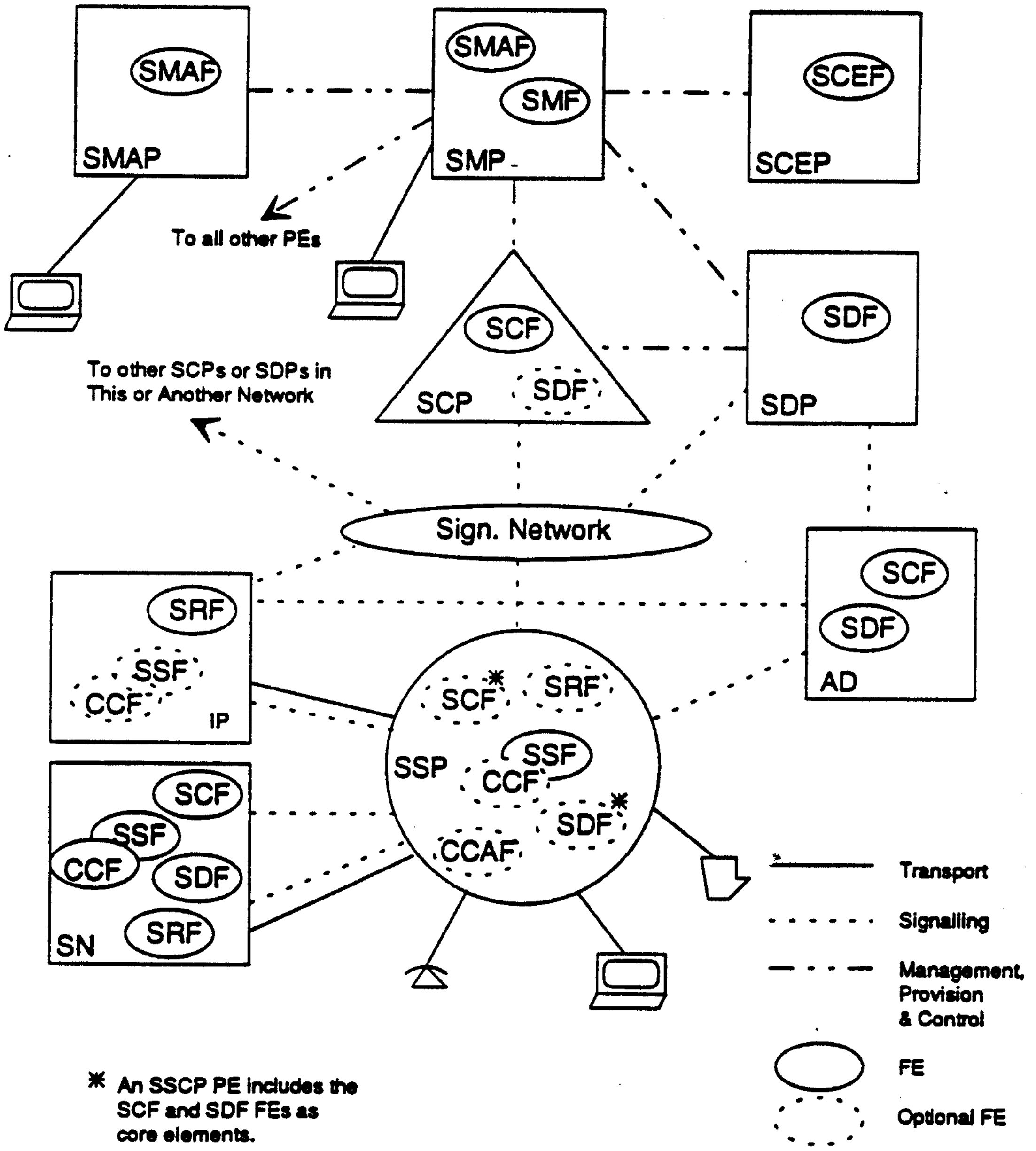


Figure 41: Scenarios for Physical Architectures.

5 Intelligent Network (IN) long term architecture framework

5.1 Introduction

The IN long term architecture framework is the structure whereby there is integration of technologies developed in other standards activities (i.e. ODP, DAF, TMN, etc.) into the intelligent network architecture. The framework will provide an open architecture that is achieved through the integration of computing information and telephony technologies.

This framework will structure target architecture around the concepts of providing a bridge to existing architecture by utilizing enabling technology, but not being restricted by today's technology.

The architecture will be enhanced by evolving service requirements and emerging technologies. The emerging technologies should include Broadband Capabilities, Distributed Processing (e.g. Distributed Data base), Open Systems Interconnection, Object-Oriented Modelling, Information Technology, Cooperative Processing, Distribution Control, Management of Services and Network, Verification/Validation and Artificial Intelligence.

5.2 Intelligent Network conceptual model

It is intended that the IN conceptual model remains consistent throughout the evolution of the IN architecture. Evolutionary changes based on experience, service requirements or technologies should be accommodated, as necessary, within this constraint.

5.3 Architecture structure

5.3.1 Logical architecture

The IN long term architecture will facilitate the development of intelligent networks. These networks will have the attributes of: Integrated Services; Integrated/Shareable Control; Programmability; Adaptability facilitated by modularized hardware and software; Interoperability of networks and systems; an OSI-aligned protocol architecture for all interfaces that facilitate communication between entities. The networks consist of interconnected nodes. Each node consists of functional groupings including: Service function; Interconnect function; and Communication function. Service function contains service logic, which includes the capability to translate customer requests into network actions necessary to execute those requests, functional components and service control programmes. The Interconnect function establishes and maintains the connections, while providing ~~the network and~~ network node interfaces. The Communication function facilitates communications among network nodes/units, among networks, and between the network and the customer. By definition, a node may provide the entire set of functions or any subset.

5.3.2 Physical architecture

Elements of the physical architecture are nodes and interfaces. With the expected unification of all protocols and the alignment of the resulting protocol with the OSI reference model, all the interfaces should have the same protocol architecture. That includes the network-to-network interface. However, this does not mean all the interfaces would be identical. It means that one would have to implement, only, those protocol messages necessary to cover the functionality at that interface. The syntax and the semantics of those messages, though, would be the same. The OSI reference model is expected to evolve to accommodate the specific needs of telecommunications.

5.3.3 Open distributed processing view

The open distributed processing view of the IN long term architecture introduces the concepts of application portability and an open system. Modelling an intelligent network or its node as an open distributed information system can be used to facilitate application portability and open communication. Application/information processing can be defined as a service. Interactions between the service and the computing system platform, on which the service runs, is required.

From a computational viewpoint, the service and the computing platform can be represented as aggregated collections of objects. An arbitrary boundary between the service and the computing platform is used to differentiate objects in the service from those provided by the network provider in the platform (see Figure 42). An object should be considered as a set of software/hardware components. The computing platform must be powerful and flexible enough to support the expected service needs. The platform's capabilities will be determined by service needs and technological opportunities.

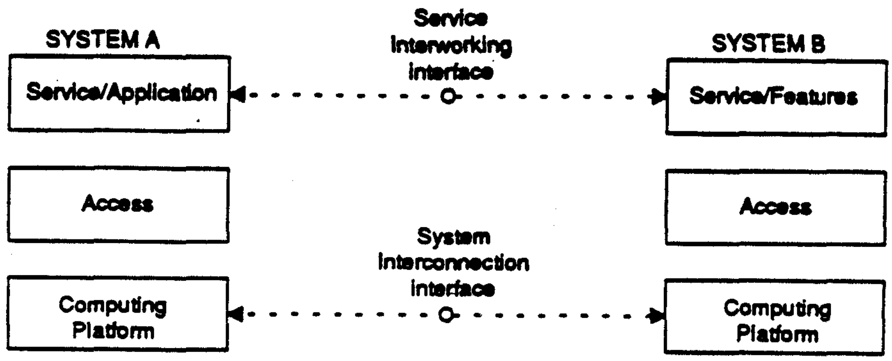


Figure 42: Open distributed processing view

Portability of distributed applications guarantees that objects executed by a system have the potential to migrate to another system without disruption of the service provided, or the services used. Migration include both the static case of reconfiguration and the reloading of a system and dynamic reconfiguration.

Open systems allow a broad range of activities to be accomplished, such as: interprocess communication; data representation; data storage; processing and resource management. Cooperation between open systems is accomplished through two abstract interfaces: service interworking interface and system interconnection interface.

Additional Open Distributed Processing Viewpoints are described in the ODP/DAF standards, each representing a different set of abstractions of the distributed system. Viewpoints are pragmatic tools, each leading to a representation of the system with emphasis on a specific concern. Within IN the notion of viewpoints has been adopted in the concepts of planes in the IN conceptual model. Multiple viewpoints must be considered to ensure portability.

Distribution transparency is a DAF mechanism that achieves evolvability and compatibility with existing technologies. Distribution transparency refers to the hiding or isolation of the effects of distribution. Transparency may be supported by providing sets of transparent objects within the infrastructure. For example, location transparency hides the location of objects that interact with it. Within IN, location transparency can be used to support subscriber portability which enables a subscriber to access a service in the same manner independent of the location from which the network is accessed. Additional transparencies relevant to IN include concurrency transparency, which hides the existence of concurrent users of an object, and replication transparency, which hides the effects of having multiple copies of objects. The integration of transparency into the design of interfaces in the IN architecture enables greater freedoms and independence in the design of applications and services. Services may be designed independent of where, within the system they implemented and independent of the physical or logical configurations of the system.

5.4 Service considerations

It is intended that the long term architecture support video, image, audio, text and data services, etc.

5.4.1 Service/service feature interaction

As part of the service definition, interaction between independently defined services will have to be taken into account. The handling of these interactions will utilize a mixture of technologies described in section 4.5 and new methods to be devised within the long term architecture framework studies.

Two types of interactions will have to be taken into account:

- interactions resulting from combined usage of service definitions;
- interactions resulting from the interactions of components of which the services are built.

(This will recursively apply for the service components.)

To handle interaction situations an identification of these situations will have to be done. Amongst others, two methods are foreseen:

- static interaction - identification resulting from common usage of resources (data)
- dynamic interaction - identification resulting from sequencing services/components

Thus, several phases can be foreseen at which interactions should be handled.

- at service definition (creation) level;
- at service deployment level;
- at service activation level;
- at service execution level.

(This can be recursively applied to components building the service.)

A mixture of methods and tools in all phases will be needed to handle the interaction problem.

Service creation, service deployment, service activation and service execution levels will require further study.

5.5 Technology basis

It is intended that the IN long term architecture take advantage of new/emerging technologies as appropriate. Several of these technologies are described in the subsequent sections 4.5.1 through 4.5.10.

Intelligence resides in all physical and logical elements of the IN network. This distributed intelligence gives the network the flexibility to respond to new requirements in an efficient and effective manner.

5.5.1 Broadband capabilities

Broadband transport, traditionally, has concentrated on service control and bearer control functionalities. The support of new capabilities in switching and transport provided by emerging new techniques (e.g. new transfer modes, different communication configurations and the separation between call and connection control) will lead to additional requirements on the IN long-term architecture.

5.5.2 Distributed processing

Distributed processing is the mechanism for maintaining a multi-functional environment, that is composed of a variety of applications, protocols and platforms. Several important issues include Scalability, Portability and Performance components.

Scalability and Portability are the means to interface with various platforms, independent of size or complexity. In the area of network services, it allows the construction of remote and local operations; utilization of information system resources via applications, where the applications are unaware of whether the interface platform is local or distributed; and that the lack of awareness should not affect their proper operation.

As networking interfaces improve, it will be possible to treat groups of applications as a single image of an application platform, known as distributed systems. A distributed system may have common objects, authentication information and software/service interaction. An important area is data communication services, which function as data transport services.

The concepts of data base management also needs to be applied to the network environment. The conventional view of data base management has been applied to data manipulation capability and; hardware and software retrieval methods. Data base services are the specialized data services required to access and manage structured elements through the management of information processing and the data base infrastructure.
Standards work in distributed processing focuses on the integration between diverse applications and across diverse technologies that include telecommunications, multimedia systems, and information services.
The seminal work in modeling of distributed systems and architecture for distributed processing within the standards arena are being done in CCITT SG VII Q19 DAF (Distributed Application Framework). DAF, in collaboration with ISO/IEC JTC1 SC21 ODP (Open Distributed Processing), is producing a number of Recommendations (to become the X.900 series of Recommendations) on a framework for standardization of distributed processing. The primary objective of these efforts is the provision of a standard infrastructure which enables distributed transparency, or the hiding of the effects of distribution from the applications. This transparency includes independence from the details of the communications mechanisms to be used.
A complete distributed processing solution addresses issues regarding interoperability among heterogeneous systems, portability of software between heterogeneous systems, integration of heterogeneous software and technologies to provide consistent solutions to future information needs, scalability of these solutions, and improved performance.

5.5.3 Open Systems Interconnection (OSI)

Communication systems which employ the standardized communication procedures and methods derived from the OSI Reference Model and interconnection are referred to as "Open Systems" and "Open Systems Interconnection" respectively. Defined procedures enable the interconnection and the subsequent effective exchange of information between users. The OSI Reference Model, in particular, will permit interworking between different networks, of the same or different types, to be defined such that communication may be achieved as easily over a combination of networks as over a single network.

OSI is concerned not only with the transfer of information between systems, (i.e. transmission), but also with their capability to interwork to achieve a common (distributed) task. In other words, OSI is concerned with the interconnection aspects of cooperation between systems, which is implied by the expression "systems interconnection".

An integral part of OSI is the concept of a layered architecture, with defined layers, entities, service-access-points, protocols and connections. Elements of layer operation include connections, transmission of data, error functions, routing aspects and management aspects. The management aspects of the OSI architecture are described in section 4.5.8 (Management of services and networks).

The resulting OSI architecture is composed of seven layers: Application layer; Presentation layer; Session layer; Transport layer; Network layer; Data link layer; and Physical layer. The functions these layers provide include: Communication between open systems; Syntax data representation; Management and synchronization of dialogue, data exchange interactions and network connections; Optimized usage of available network services. The network connections range from point-to-point configurations to complex combinations of sub-networks with different characteristics. The data link connections provide the data circuit interconnection (i.e. communication paths in the physical media).

5.5.4 Object-oriented modelling

The use of Object modelling could satisfy the modelling needs of IN long term architecture by the use of abstract object modelling concepts. These concepts are utilized within, both, the TMN (Telecommunications Management Network) work and the DAF (Framework for the Distribution of Applications) work.

Generic network modelling for TMN is a common set of principles that utilize generally accepted modelling guidelines which facilitate the process of defining and selecting interface protocols. DAF has defined an abstract object model which provides a formal framework.

The object-oriented methodology allows strict modular system specifications. The strong encapsulation supported by object-oriented design is a prerequisite for openness to evolutionary changes within a specification. The realization of the potential benefits that can be achieved by the user of object-oriented analysis and design has resulted in the adoption of this approach in various domestic and international standard arenas.

To ensure the tractability of large distributed systems, they must be designed in a manner that minimizes the interdependence of components of the system. Object-oriented modelling techniques are used because they provide the benefits of abstraction, encapsulation and modularity. Abstraction is a tool for simplifying system descriptions by describing characteristics that are meaningful to users while suppressing the irrelevant characteristics. Encapsulation is a technique for only exposing the observable behavior of an object's services and providing clients with information about how to invoke these services while hiding the details of the object's implementation. Modularity is achieved because object models permit the specification of a system as an aggregation of collections of objects. Thus, system descriptions are simplified by allowing subsystems to be treated as independent objects. These tools helps reduce the complexity of a system when looking at it from a particular point of view.

The ability to re-use specifications, a virtue in a non-distributed environment, becomes a necessity in a highly distributed one. Specifications must therefore be highly modular and knowledge of components must be restricted as much as possible. Object modelling promotes modularity by enabling the structuring of specifications into smaller parts. By constraining all object interactions to take place at well defined interfaces, the interdependence of objects is minimized. This provides a basis for reusability and adaptability. By isolating and explicitly describing all object interactions, the dynamic and evolutionary nature of distributed systems can be more easily modelled.

The object-oriented design philosophy enforces a discipline and precision of specification that is required for the description of complex, distributed systems. For the IN long term architecture, there is the need to outline an abstract object model and to begin using this paradigm for the description of Objects and interfaces required in the IN long term architecture.

5.5.5 Information technology

Information technology provides the means for an enterprise to integrate, share, process and distribute large quantities of information in order to meet the needs of a wide variety of internal and external customers. System design, software development, and operation procedures are important aspects of this technology. In the context of IN, this technology would initiate Transport services; Operation, Administration, Maintenance and Provisioning Services (OAM and P); and Customer Data Interactions.

5.5.5.1 Software/programming assistance

Software tools, techniques, languages and procedures should play a major role in areas such as Service Interaction. Independent components of the Service Interaction may be maintained or initiated by a specific software tool/procedure or a set/series of tools/procedures.

Computer aided software engineering (CASE) technologies enable the management of the entire software development process, from requirements to design, coding, testing, and maintenance. Use of CASE technology is expected to reduce the incremental cost of new application development and to enable new applications to be developed faster.

CASE tools are used to make products that are stored in a CASE data base repository, which provides support for the storage, retrieval, version management, and configuration management of these products and the relationships among them. CASE integrates these various products to ensure consistency among them and permit reuse at each stage. The repository supports distributed access to these software products and heterogeneous and distributed storage. CASE technology enables the speedy development of new applications by encouraging the practice of reuse. Use of CASE is aimed at achieving an improvement in quality and improving the management and productivity of the process. Existing and new CASE technologies and techniques can be applied within IN to address the dynamic aspect of service creation. The use of CASE technologies within IN will enable the rapid creation and activation of new application components.

5.5.6 Cooperative processing

Cooperative processing is a specialization of distributed processing across two systems. Cooperative processing is the means by which, both data and network functionality can process cohesively within the processing structure. This is accomplished by the introduction of an operational control mechanism. The operational control components include Node Connection Control, Global Connection Control and Service Control. Node connection is concerned with the manner in which a connection is established through an individual network node to a specific link which leads to another node (i.e.circuit-based or packet-based connection). Global connection is concerned with the manner in which one or more end-to-end connections through multiple nodes is established and monitored in the network. Service control provides the ability to effectively define and ~~customize~~ a service.

The control components of IN consists of those functions and activities through which distributed network intelligence is coordinated, administered, and utilized. The control intelligence is itself distributed throughout the network to allow for the necessary scalability and performance. New services which utilize the intelligence throughout the network will require a more robust control model.

The availability of remote data base(s) will allow the introduction of new services, which will need access to large amounts of volatile data. Therefore, the types of services that can be defined and employed in the network could be easily expanded. Remote data base(s) may provide "intelligence", in the sense that the specific data returned as a result of a retrieval request may depend on logic embedded in the data. This is not "traditional" network control functionality, in the sense that something other than the requested information may be obtained. It may, however, be considered to be a part of service control. If the data base logic is used for only one service then this control can be considered to be explicit. If it is used for multiple services, then the control will become implicit. In either case, most of the control will still be embedded in and integrated with the node processing function.

Another example of cooperative processing is the utilization of "intelligent terminals", e.g. a workstation consisting of a "smart card reader", which performs some of the network tasks (i.e. authentication and authorization).

5.5.7 Distributed control

A major part of distributed control is the allocation, control and management of resources. Certain types of resources can be centralized and other types can be distributed. Distributed control makes the availability of resources more effective. Connection to alternative pools of resources would also be possible. The function of reserving resources is also important. There could be a need of a specific resource type for certain services to be reserved before establishing a connection or replying to an application request.

Distributed control also provides the framework for representing and handling service/feature interactions.

High-level Networking Services provide an application with a very high-level interface to networking capabilities. These services do not require the application to be aware of any of the low-level network details.

Simple Networking Services are designed to be used to provide applications with an interface to simple application-to-application communication over a network. This interface will be used in applications that perform explicit connection-oriented or connectionless networking activity, but do not want to control the low-level details of the networking interface.

5.5.8 Management of services and networks

The management aspects of IN can be summarized as the management of Service and Network generic functions, along with IN specific managed objects. Managed objects are very diverse. They can be divided according to their logical, functional and physical nature. Examples of managed objects include: UPT Service, Service Independent Building Blocks, Trigger Table Data, Network Services (i.e. LAN, ISDN), Transmission Services, Network operator role, OSI application process, OSI resources, Virtual Circuits, Connections and calls.

To manage all the different kinds of managed objects an integration of the IN architecture and TMN concepts should be accomplished. One of the outcomes of this integration will be the identification of IN specific management requirements concerning the deployment, provisioning, control, monitoring and billing/charging of services. These requirements contain amongst others configuration, performance, fault, accounting and security aspects.

The network environment needs to manage physical connections; network protocols and formats, and distributed systems services.

Within the OSI architecture, there is a need to recognize the special problems of initiating, terminating, and monitoring activities and assisting in their harmonious operations, as well as handling abnormal conditions. These have been collectively considered as the management aspects of the OSI architecture. The defined categories of management activities include application management, systems management and layer management.

Because of the vast programmability possibilities of new services, the IN long term architecture should also support the ability to programme (in parallel with the service processing logic) management logic. To accomplish this, Management SIBs may be provided.

5.5.9 Verification/validation

The function of verification/validation is to insure that features/service interactions perform as specified. This may include testing of network performance to ensure consistency with quality of service and network integrity objectives. This can be accomplished by providing complimentary verification/validation standards to qualify functional performance and consistency.

5.5.10 Artificial intelligence

Artificial intelligence is the means by which IN will evolve the existing network base with the latest emerging technologies. The concepts utilized may involve non-traditional network elements. This should facilitate a wide variety of services and interfaces (i.e. user-to-user, user-to-network, network-to-network, etc.).

6 Vocabulary of Terms Used in the Definition of Intelligent Networks

6.1 General

This Recommendation provides a vocabulary of terms and definitions which have been studied for application in the documentation of Intelligent Networks.

6.2 Terms and definitions (listed alpabetically)

Access	A means of interaction between a user and a network.
Access channel	A designated part of the information transfer capability having specified characteristics, provided at the user-network interface.
Access function	A set of processes in a network that provide for interaction between the user and a network.
Application	An entity in a network that provides a user with a set of capabilities that have an implementation - independent specification, to control the communication processing, and storage of information and to control a user interface.
Application Entity (AE)	The aspects of an application process pertinent to OSI.
Application program (IN)	Logic residing in the Service Control and Service Management realms that directs and/or controls the performance of actions in the network to provide and/or manage the provision of IN service features.
Application Programming Interfaces (APIs)	A set of interfaces between the application realm (e.g. service control and service management) and the network realm.
Application Service Element (ASE)	A part of an application entity which provides OSI environment capability, using underlying (OSI) services where appropriate.
Architecture	Any ordered arrangement of the parts of a system.
Association	A logical relationship between entities to perform a function.
Attribute	An intrinsic property of an object.
Basic call	A call between two users that does not include additional features (e.g. a plain telephone call).
Basic Call Model (BCM)	A description of the sequence of activities used in processing a basic call attempt.
Basic Call State Model (BCSM)	A high-level finite state machine model of call processing for basic call control (i.e., a two party non-IN call). The model might only cover a portion of a call attempt, e.g. an originating BCSM or a terminating BCSM, or the entire call connection, originating user to terminating user.
Call	The use, or possible use, of one or more connections set up between two or more users and/or service(s).
Call Control	The set of functions used to process a call (e.g. provide service features and establish, supervise, maintain and release connections).
Call Control Agent Functional Entity (CCAF)	A functional entity that provides network access functions for users, interacting with Call Control Functional entities in providing services.

Call Control Funtional entity (CCF)	Functional entities which cooperate with each other to provide network call processing functions.
Call model	A representation of functions involved in processing a call.
Call Processing State Model (CPSM)	A finite state machine model of call processing for Intelligent Network call control.
Call/service processing	The execution of logic by a switching or control function to advance a call attempt or service request.
Capability Set (CS)	A set of Intelligent Network capabilities that are to be the subjects of standardization activities and for which the availability of standards recommendations will be targeted for a particular time frame.
Connection	An association of transmission channels or circuits, switching and other functional units set up to provide a means for a transfer of information between two or more points in a telecommunications network.
Connection control	The set of functions used for setting up, maintaining and releasing a communication path between two or more users or a user and a network entity, e.g. a dual tone multifrequency receiver.
Connection segment	A representation of a specific portion of a call/service attempt.
Connection Segment Model (CSM)	A representation of a total connection in terms of connection segments.
Control	To exercise a directing influence.
Control window	The interval from the generation of an initiating information flow, e.g. "Request Instructions", until an information flow that terminates a control relationship has been received.
Core feature	A service feature that is fundamental to a service, i.e. in the absence of the feature, the service is not a viable service offering.
Data	User and/or network information stored in the network used in connection with call/service processing. An instance of a data object.
Data base	A physical entity used for storing information.
Data management	Establishing, updating and administering data bases in the network.
Data object	An individually addressable unit of information specified in a data template.
Data template	A specified logical structure for a collection of data objects, including allowable ranges for their values and other data consistency specifications.
Detection point	A point in basic call processing at which a processing event may be reported to the Service Control Function and transfer of control between SSF/CCF and SCF can occur.
Dialog(ue)	A conversation or an exchange of information.
Directory	A listing of objects in a specific sequence.
Distributed Functional Plane	The plane in the Intelligent Network conceptual model containing functional entities and their relationships.
Dynamic arming	Enabling of a detection point by a Service Control Function in the course of service control execution for a particular call/service attempt. The detection point remains armed only until the event is detected or service control ends.
Dynamic data	Information subject to change as a result of call/service processing.

Element	An identifiable physical unit.
Elementary function	A primary or basic function that cannot be further decomposed.
Entity	A part, device, subsystem, functional unit, equipment or system that can be individually considered. In ISDN the term is used to refer to a particular system or sub-system such as a user terminal or a digital exchange. It is also used to refer to a set of functions of a particular system at a location, e.g., the layer 2 functions of a signalling system at a user terminal.
Event	A specific input to and/or output from a given state in a finite state machine model that causes a transition from one state to another.
Event Detection Point (EDP)	A detection point that is dynamically armed.
Executive process	A process that controls the execution of other processes.
Feature	A reusable capability provided to a user by one or more services in a network.
Feature interaction	A situation that occurs when an action of one feature affects an action or capability of another.
Finite State Machine (FSM)	A system having a finite number of states and specified transitions between states.
Finite state machine model	An operational model of an entity that is described by the finite set of states the entity can be in and the finite set of transitions possible from one state to another.
Function	A set of processes defined for the purpose of achieving a specified objective.
Functional entity	A grouping of service providing functions in a single location and a subset of the total set of functions required to provide the service.
Funtional Entity Action (FEA)	An action performed in a functional entity as a result of a specific stimulus while the functional entity is in a specific state. At the conclusion of the action the functional entity waits for another stimulus.
Functional routine	Logic that controls the performance of set of actions to accomplish "routine" tasks, e.g. retrieve information, pass information, etc.
Global control	Control of a process whose functions are distributed among several entities.
Global function	A set of elementary functions in the global functional plane of the Intelligent Network conceptual model required to provide a service or an aspect of a service.
Global Functional Plane	The plane in the Intelligent Network conceptual model which defines Service Independent Building Blocks (SIBs) used in providing service features.
Independent or independence	Not necessarily specific to one aspect.
Information flow	An interaction between a communicating pair of functional entities.
Intelligent Network (IN)	A telecommunications network architecture that provides flexibility for facilitating the introduction of new capabilities and services, including those under customer control.
Intelligent Network Application Protocol (INAP)	A protocol for Intelligent Network applications contained in layer 7 (application of the OSI model).

IN conceptual model	A planning model used for defining the Intelligent Network architecture.
IN Data Base (INDB)	A physical entity used for information storage in the Intelligent Network.
IN Data Base Management System (INDBMS)	A system used for establishing and/or administering information storage in the Intelligent Network.
IN Funtional Architecture (INFA)	An Intelligent Network Functional Architecture is used to achieve independence of services from logical resources, and in turn logical resources from physical resources. It provides the user of the IN with a consistent view of the services independent of how the logical resources may be assigned in order to provide services.
IN service	A service provided using the capabilities of the Intelligent Network.
IN structured network	A network incorporating the IN concept.
Intelligent peripheral	A physical entity that implements the Intelligent Network specialized resource function.
Interface	A shared boundary or interconnection between two entities or two devices.
Layer	A conceptual region that embodies one or more functions between an upper and a lower logical boundary within a hierarchy of functions.
Leg	A representation within a call processing state model representing a telecommunication path towards some addressable entity (e.g. a path toward a user, intelligent peripheral unit, etc.)
Library	An assembly of objects, routines, programs, etc. that may be drawn upon for use in the performance of functions.
Manager	A function that directs and/or controls operations of a function or an assembly of functions to allow a functional entity to perform all or a part of the expected functional entity actions.
Management	The function of directing, maintaining and/or administering.
Management function	A set of processes used for the management of an entity (e.g. data base management).
Management Building Block (MSIB)	A reusable set of functional entity actions and information flows used to provide or support service management functions in the network.
Network Functional Architecture (NFA)	The Network Functional Architecture consists of functional groupings of sets of functions. It maintain independence of service from logical resources, and that of resources from physical resources.
Network implementation independence	Not dependent on a specific network configuration.
Network interworking	A process in which different networks cooperate to provide a service.
Network management	An activity to support the proper operation of an IN-structured network.
Network Operator (NO)	The Network Operator has the responsibility of operating elements of the network, which are used to support the service operations.
Object	An intrinsic component of an entity that is described at an appropriate level of abstraction in terms of its attributes and functions.
Optional feature	A service feature added to core features to optionally enhance a service offering.

Persistent data	Information that endures beyond a single instance of use, e.g. longer than or call attempt.
Physical plane	The plane in the Intelligent Network conceptual model containing elements and their interfaces that implement functional entities.
Plane	Part of A layer in the Intelligent Network conceptual model.
Point in call	A state in a basic connection state model.
Point of initiation	A functional interface between basic call processing and service logic over which service control is initiated.
Point of return	A functional interface between service logic and basic call processing over which call processing control is returned to basic call processing.
Relationship	The complete set of information flows, where they exist, between two functional entities.
Resource	In telecommunications, any network element that can be drawn upon in providing service, e.g. a circuit, a receiver, etc.
	A service is a stand-alone commercial offering, characterised by one or more core service features, and can be optionally enhanced by other service features.
Service control	Direction of the functions or processes used to provide a specific telecommunications service.
Service Control Function (SCF)	The application of service logic to control functional entities in providing Intelligent Network services.
Service Control Point (SCP)	An entity in the Intelligent Network that implements a service control function.
Service creation	An activity whereby the capability to provide a supplementary service is brought into being from specification to development and verification.
Service Creation Environment Function (SCEF)	The set of functions that support the service creation process, the output of which includes both service logic programs and service data.
Service Creation Environment Point (SCEP)	A physical entity that implements the service creation environment function.
Service creation platform	A set of service independent objects or functions which allow the creation of services in an Intelligent Network.
Service creation process	The conception, design and implementation of a capability to provide a service.
Service data	Customer and/or network information required for the proper functioning of a service.
Service Data Function (SDF)	The set of functions that provides for the management of service data in accordance with a service data template.
Service Data Point (SDP)	A physical entity that implements a service data function.
Service data template	A data template related to a specific service logic program.

	A service feature is a specific aspect of a service that can also be used in conjunction with other services/service features as part of a commercial offering. It is either a core part of a service or an optional part offered as an enhancement to a service.
Service independence	Not necessarily specific to one service.
Service independent	(1) Not dependent on the availability of other services or (2) Freedom to create any service desired.
Service Independent Building Block (SIB)	A reusable set of functional entity actions and information flows used to provide a service feature or a part of a service feature in an Intelligent Network.
Service logic	A sequence of processes/functions used to provide a specific service.
Service Logic Program (SLP)	A software program containing service logic.
Service Logic Program (use) Instance (SLPI)	The invocation and application of a particular service logic program in providing a service or a service feature for a specific call/service attempt.
	It is an activity to support the proper operation of a service and the administration of information relating to the user/customer and/or the netwo operator. Service management can support the service control, billing and service monitoring.
Service Management Access Function (SMAF)	A functional interface between network operators and/or subscribers and network service management functional entities.
Service Management Function (SMF)	The set of processes that support the management of user and/or network information, including service data and service logic programs that are required for the proper operation of a service.
Service Management Point (SMP)	A physical entity that implements a service management function.
Service Management System (SMS)	An entity usually external to the network that is used in managing customer and/or network data bases.
Service plane	The plane in the Intelligent Network conceptual model that contains service service entities and their relationships.
Service processing	The execution of service control and basic call processing functions to provide a service.
Service provider	An organization that commercially manages services offered to service subscribers.
Service subscriber	An entity that contracts for services offered by service providers.
Service Switching Function (SSF)	The set of processes that provide for interaction between a call control function and a service control function.
Service Switching Point (SSP)	A physical entity that implements a service switching function.
Service user	An entity external to the network that uses its services.
Single-ended service feature	A feature, e.g. call/service attempt manipulation, that applies to a only one o the parties that may be involved on a call/service attempt.

Single point of control	A control relationship where the same phase or aspect of a call/service attempt is influenced by one and only one Service Control Function.
Specialized Resource Function (SRF)	The set of functions that provide for the control and access to resources us in providing-services in the Intelligent Network.
State (in FSM)	A description of an entity defined by the values of its object attributes at a given point in time.
State (in SDL)	A condition in which the action of a process is suspended awaiting an input.
Static arming	Enabling of a detection point, as directed by a Service Management Functio (SMF), to cause a specified action by call/service processing whenever a specific point in call/service processing is encountered. The detection point remains armed until it is disarmed as directed by the SMF.
Static data	Information that remains unchanged for the duration of a call or incident of use of a service. (Usually controlled by a source external to the network).
Supplemented call	A basic call with added service features or capabilities.
Terminator segment	A connection segment that shows elements of a connection between a network access point and an entity containing Service Switching/Call Contro functions.
Transition	In a finite state machine model, a change in the state of an entity resulting from a change in the values of its object attributes.
Trigger	A stimulus for initiating an action.
Trigger Detection Point (TDP)	A detection point that is statically armed.
User	An entity external to the network that uses its service(s).
Vendor or implementation independent	Products from different vendors are able to work together in the same environment, and/or, physical units serving as the same functional entity(ies) produced by different vendors' can be used interchangeably.
Work station	A physical entity that implements the work station function.
Work Station Function (WSF)	Processes that allow communication between humans and a system.

History

Document history	
March 1992	First Edition Approved by TTC 10
March 1996	Converted into Adobe Acrobat Portable Document Format (PDF)