INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Q.65
(06/97)

SERIES Q: SWITCHING AND SIGNALLING

Functions and information flows for services in the ISDN – Methodology

---

The unified functional methodology for the characterization of services and network capabilities

ITU-T Recommendation Q.65

(Previously CCITT Recommendation)

ITU-T Q-SERIES RECOMMENDATIONS

SWITCHING AND SIGNALLING

SIGNALLING IN THE INTERNATIONAL MANUAL SERVICE	Q.1–Q.3
INTERNATIONAL AUTOMATIC AND SEMI-AUTOMATIC WORKING	Q.4–Q.59
FUNCTIONS AND INFORMATION FLOWS FOR SERVICES IN THE ISDN	Q.60–Q.99
Methodology	**Q.60–Q.67**
Basic services	Q.68–Q.79
Supplementary services	Q.80–Q.99
CLAUSES APPLICABLE TO ITU-T STANDARD SYSTEMS	Q.100–Q.119
SPECIFICATIONS OF SIGNALLING SYSTEMS No. 4 AND No. 5	Q.120–Q.249
SPECIFICATIONS OF SIGNALLING SYSTEM No. 6	Q.250–Q.309
SPECIFICATIONS OF SIGNALLING SYSTEM R1	Q.310–Q.399
SPECIFICATIONS OF SIGNALLING SYSTEM R2	Q.400–Q.499
DIGITAL EXCHANGES	Q.500–Q.599
INTERWORKING OF SIGNALLING SYSTEMS	Q.600–Q.699
SPECIFICATIONS OF SIGNALLING SYSTEM No. 7	Q.700–Q.849
DIGITAL SUBSCRIBER SIGNALLING SYSTEM No. 1	Q.850–Q.999
PUBLIC LAND MOBILE NETWORK	Q.1000–Q.1099
INTERWORKING WITH SATELLITE MOBILE SYSTEMS	Q.1100–Q.1199
INTELLIGENT NETWORK	Q.1200–Q.1999
BROADBAND ISDN	Q.2000–Q.2999

For further details, please refer to ITU-T List of Recommendations.

ITU-T RECOMMENDATION Q.65

THE UNIFIED FUNCTIONAL METHODOLOGY FOR THE CHARACTERIZATION OF SERVICES AND NETWORK CAPABILITIES

Summary

This Recommendation contains the Unified Functional Methodology (UFM), describing a common functional architecture for providing services and addressing signalling requirements for service implementation. The overall method for deriving switching and signalling Recommendations for ISDN services, consisting of three stages, is described in Recommendation I.130. The method has been generalized beyond ISDN to include services provided in and among networks of various types. The UFM combines the traditional approach of the 1988 version of this Recommendation with some of the approaches traditionally used in the Intelligent Network (IN) description method. The detailed method for deriving the Stage 2 portion of these Recommendations is described in this Recommendation. The main text of this Recommendation presents background information on the unified functional methodology, the steps of the method, the conventions used in description techniques, and guidelines for usage.

Source

ITU-T Recommendation Q.65 was revised by ITU-T Study Group 11 (1997-2000) and was approved under the WTSC Resolution No. 1 procedure on the 5th of June 1997.

FOREWORD

ITU (International Telecommunication Union) is the United Nations Specialized Agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of the ITU. The ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Conference (WTSC), which meets every four years, establishes the topics for study by the ITU-T Study Groups which, in their turn, produce Recommendations on these topics.

The approval of Recommendations by the Members of the ITU-T is covered by the procedure laid down in WTSC Resolution No. 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

INTELLECTUAL PROPERTY RIGHTS

The ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. The ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, the ITU had/had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

CONTENTS

Recommendation Q.65

THE UNIFIED FUNCTIONAL METHODOLOGY FOR THE CHARACTERIZATION OF SERVICES AND NETWORK CAPABILITIES

(revised in 1997)

1 Introduction

This Recommendation contains a summary of the Unified Functional Methodology (UFM), describing a common functional architecture for providing services and addressing signalling requirements for service implementation. The overall method for deriving switching and signalling Recommendations for ISDN services, consisting of three stages, is described in Recommendation I.130. The method has been generalized beyond ISDN to include services provided in and among networks of various types. The UFM combines the traditional approach of the 1988 version of this Recommendation with some of the approaches traditionally used in the Intelligent Network (IN) description method. The detailed method for deriving the Stage 2 portion of these Recommendations is described in this Recommendation.

The main text of this Recommendation presents background information on the unified functional methodology, the steps of the method, the conventions used in description techniques, and guidelines for usage. Appendix I presents an outline for drafting a Stage 2 service description.

1.1 Unified Functional Methodology (UFM): Summary

The Unified Functional Methodology (UFM) allows for functional descriptions of services with information flows, specification description language (SDL), and Functional Entity Actions (FEA) starting from a single unified functional architecture. The concept of Service Independent Building Blocks (SIB) has been adopted to address service creation needs as well as to introduce re-usable blocks of flows, SDL, and FEAs which can be catalogued. The unified model enables all network architectures (i.e. ISDN, B-ISDN, IN, FPLMTS, and TMN) to be described in a similar manner.

The methodology needs to address current needs, but must also evolve to include improvements and new technology. The method includes many elements of the 1988 version of this Recommendation, so it can be used immediately, even as it evolves. The addition of the SIB concept is introduced to streamline the work of Stage 2 service definition. New SIBs will be created as needed, and a SIB catalogue will be maintained as a reference for service creation and definition.

Principles defining the scope of UFM include:

1) The unified functional methodology allows creation of functional descriptions where Stage 1 service descriptions are available or where network capabilities are specified for service creation.

2) The methodology is based mainly on experience from ISDN Stage 2 methodology [Recommendation Q.65 (1988)] and IN methodology [Recommendation Q.1210, (1995)].

   - Recommendation Q.65 provides much background in service description utilizing a functional model, functional entities, and functional entity actions.
   - The IN method provides the flexibility, service independence, and re-use characteristics which are desirable.

3) The method is based on a unified functional model.

- The unified functional architecture may be expanded to include new requirements.
- For each service description, an appropriate set of functional entities is selected to compose the model.
- This leads to a consistent set of flows and SDLs.

4) Maintain a library of SIBs (with corresponding information flows and SDL) which are used to determine the functional architecture in the distributed functional plane (see Figure 1).

- SIB sets are being created to support the capability sets in IN Recommendations.
- SIBs are mapped onto the functional architecture complete with pre-defined functional entity actions, pre-defined SDL, and pre-defined information flows.
- SIBs can be useful tools for Stage 2 creation; other aspects (e.g. SDL descriptions, information flows) can potentially be automated.

5) The methodology addresses current needs, but must evolve to include improvements and new technology.

- Method can be used even as it evolves.
- Phased approach for use across the ITU-T, using the unified architecture.
- New SIBs will be needed (existing services; new target services).
- New relationships need to be defined.
- New target services must be considered (e.g. multiparty calls).
- Not yet an automated process, but re-use plays a major role.
- New expertise must be developed for describing services from SIBs.
- Additional verification of existing services will be useful (for accuracy of the method and learning curve).

6) The unified methodology for creating Stage 2 description should include the following:

- identification of service or network capability;
- functional model and functional entity definition (Unified Functional Model);
- SIB identification (optional);
- functional entity actions – definition;
- information flows;
- SDL (Dynamic Description) (optional) ;
- scenarios (physical allocation of functional entities).

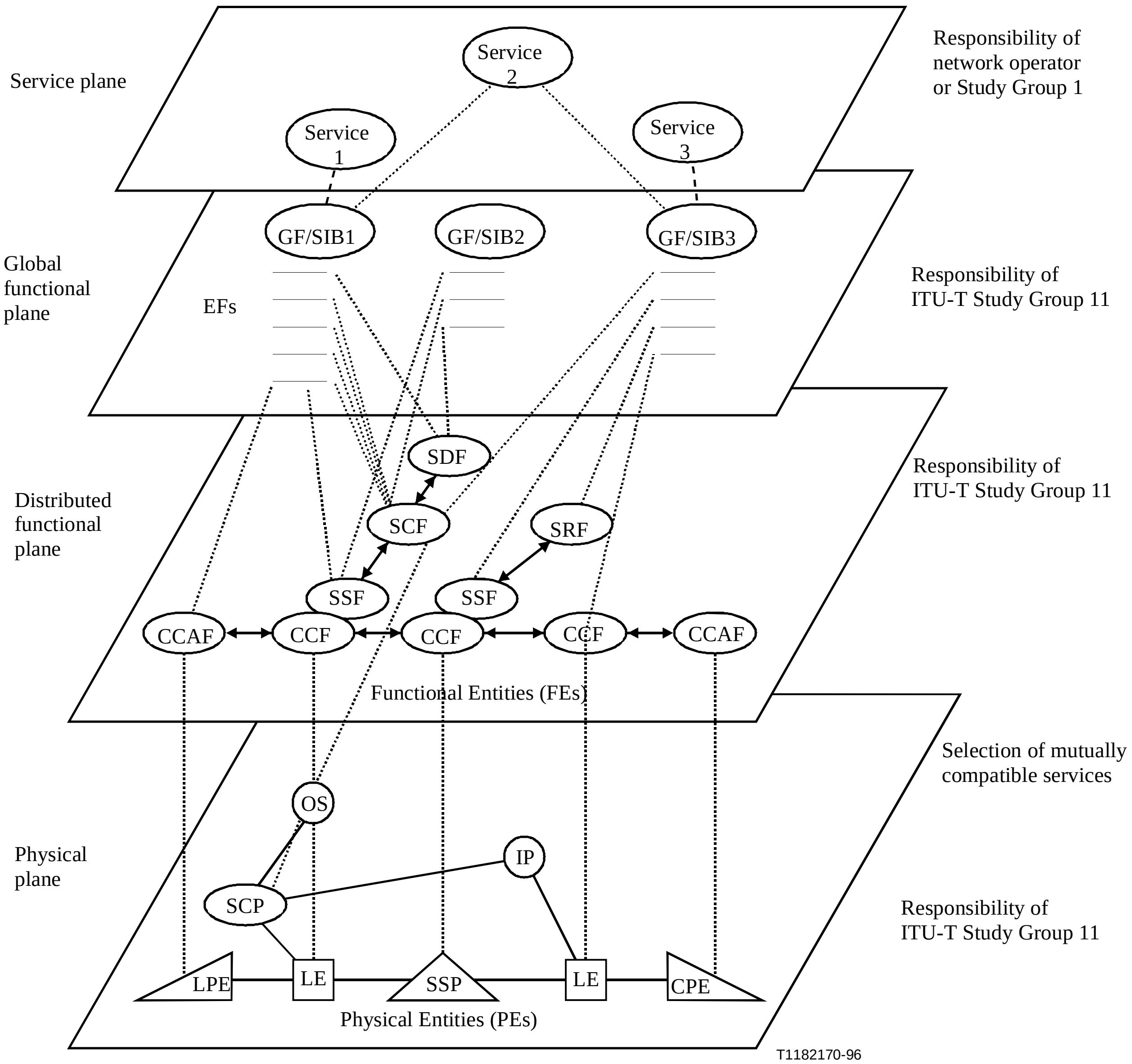
7) Define physical scenarios at an appropriate time.

- Decisions must be made as to the recommended network implementations.
- With extension to IN, B-ISDN, FPLMTS, and other networks, the catalogue of physical entities is increased.
- This greatly increases the number of potential scenarios.

8) Work to provide Recommendations for tool improvements and automation of the functional description process.

- Recommendations Z.100 (SDL) and Z.120 (Message Sequence Chart – information flows) can be utilized more effectively.

- Tools can provide validation of SDL and flows.
- Communicate rules and conventions for SIB definition and combination; some automation of this process may be possible.



.....	Mapping	GF/SIB	Global Function/Service Independent Building Block
< >	Relationship (described by information flows)	SCP	Service Control Point
——	Physical connection	EF	Elementary Function (described by FEAs)
		IP	Intelligent Peripheral

NOTE – The mappings shown are examples only; they are not intended to represent any particular service.

Figure 1/Q.65 – Conceptual methodology model showing relationships between services, global functions/service-independent building blocks, elementary functions, functional entities and physical entities

1.2 The Stage 2 definition (Summary)

– specifies a functional model using Functional Entities (FEs) from the unified functional model for a specific service or network capability description;

– specifies the functional entity actions (FEAs) needed;

– specifies information flows between FEs;

– specifies the SDL description of each FE;

– recommends a small set of realistic scenarios for the allocation of FEs to physical entities.

For services and network capabilities to be standardized, Stage 2 of the method takes as its input the Stage 1 descriptions for basic and supplementary services and network capabilities. For services and network capabilities without a detailed Stage 1 description, the Stage 2 method can utilize SIB descriptions of service features as input. The Stage 1 description views the network (this term, in this context, could include some capability in the user equipment) as a single entity which provides these services to the user. The Stage 2 description defines the functions required and their distribution within the network. The Stage 1 user-network interactions are used and interpreted within Stage 2, as illustrated in Figure 2.

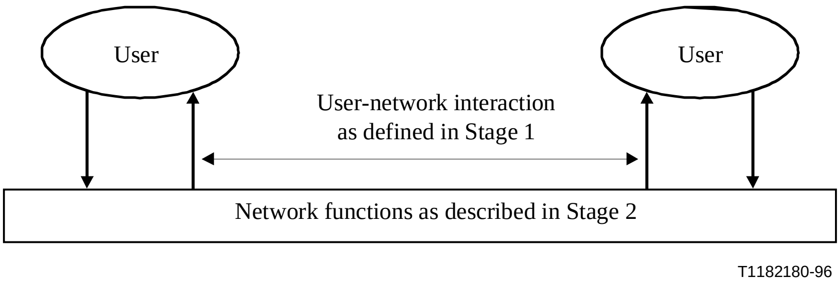


Figure 2/Q.65 – Stage 1/Stage 2 relationship

Stage 2 of the method employs techniques that provide the following desirable characteristics:

– a single functional specification which can be applied in a number of different physical realizations for providing the service;

– a precise definition of functional capabilities and their possible distribution in network equipment (and in some cases, user equipment) to support basic and supplementary services and network capabilities;

– a detailed description of what functions and information flows are to be provided, but not how they are to be implemented;

– requirements for protocol and switching capabilities as input to Stage 3 of the method.

The output of Stage 2 is used by:

a) protocol designers in Stage 3 to specify the protocols to be used between separate physical entities;

b) switch (and other node) designers to specify the functional requirements of those nodes; and

c) network planners.

This Recommendation describes the six steps of Stage 2 in detail. The order of these steps represents an idealized application of the method; however, in practice there will of necessity be iterations to define fully the Stage 2 outputs. Appendix I contains an outline for a Stage 2 description utilizing the unified functional methodology.

2 Steps of the method

2.1 Step 1 – Functional model

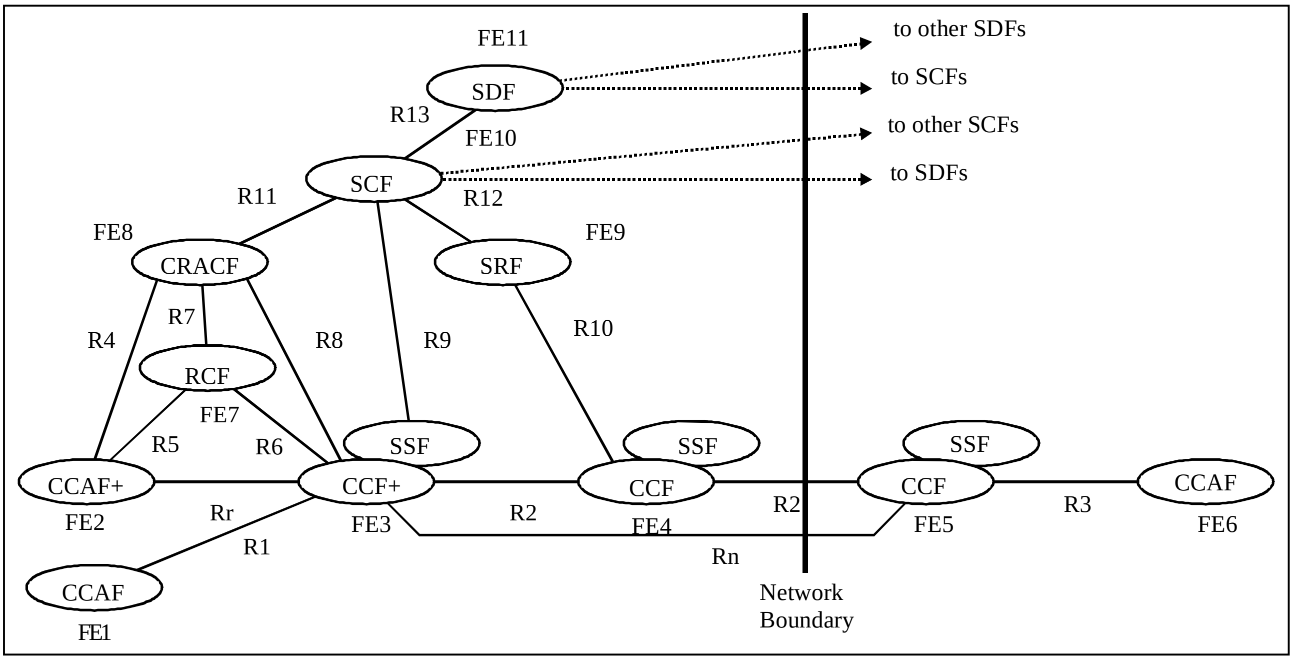
A functional model is derived for each basic service, supplementary service, or network capability. In each case the model is matched to the requirements and characteristics of the service concerned. The functional model used in the Stage 2 description of a service identifies functional entities and the relationships between them. The refinement of the initial functional model is carried out by development and/or iteration of Steps 2 to 6, as described below. The final functional model represents a result of the completed Stage 2.

2.1.1 Unified functional model

The unified functional model for the Stage 2 methodology takes account of the common functional entities, relationships, and information flows arising from ISDN, IN, B-ISDN, FPLMTS, and TMN.

Figure 3 shows an instance of the unified functional model derived from the set of FEs specified by the unified functional methodology. The functional model identifies and names the individual FEs and their types. It also identifies the relationship and relationship types between communicating FEs. Functional entities are represented by circles and the relationship between two communicating FEs is identified by a line joining them.

NOTE – The model in Figure 3 is a composite model based on basic call (Q.71), IN CS-2, FPLMTS, and B-ISDN. This model is presented as a basis from which a specific Stage 2 functional model can be constructed to support a particular service or network capability. Appendix II represents a current view on the integration of IN and B-ISDN with full separation of call and connection control. This type of model will form the basis for the evolution of this Recommendation.



CCAF	Call Control Agent Function	CCF	Call Control Function
SSF	Service Switching Function	SCF	Service Control Function
SDF	Service Data Function	SRF	Specialized Resource Function
RCF	Radio Control Function	CRACF	Call/Connection Radio Access Control Function

NOTE 1 – FE1, FE2, etc. are functional entities to meet the requirements of the particular service or network capability considered. The functional entity type is contained within the circle. Each FE is given a unique lable (e.g. FE1, FE2).

NOTE 2 – R1, R2, etc. are types of relationships between communicating pairs of functional entities.

NOTE 3 – This figure illustrates the following points:

a) A functional model may include more than one FE of the same type, e.g. CCF.
b) A functional model may include more than one relationship of the same type, e.g. R2.
c) An extension to a FE does not modify its type of relationship with adjacent FEs, e.g. R2.
d) R2 identifies a link-by-link relationship via a transit CCF.
e) Rn identifies an edge-to-edge relationship directly between serving CCFs.

Figure 3/Q.65 – Unified functional model

2.1.2 Functional entities

Functional entities are initially derived from an overall understanding of the network functions needed to support the service. Functional entities are defined as follows:

– A functional entity is a grouping of service providing functions in a single location and is a subset of the total set of functions required to provide service.

– A functional entity is described in terms of the control of one instance of a service (e.g. one call or one connection).

– A functional entity is visible to other functional entities that need to communicate with it to provide a service (i.e. functional entities are network-addressable entities).

– A functional model may contain functional entities of different types. The type of a functional entity is characterized by the particular grouping of functions of which it is composed. Thus, two or more functional entities are said to be of the same type if they consist of the same grouping of functions.

–	A separate functional entity type is normally defined for each different grouping of functions that may be distributed to separate physical devices. However, where there is a high degree of commonality between different required groupings, it may be convenient to define them as subsets of a single type rather than as different types.

–	Functional entities are derived for each basic and supplementary service or network capability. The same functional entity type may occur more than once in a functional model and also may appear in the model of more than one service.

The following are definitions of the FEs used in the unified functional model. Many of these FEs have been derived from the IN studies of the support services in the network.

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. CCAF+ includes a specific reference to the wireless environment. CCAF+ is the interface between the wireless user/terminal 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. set-up, 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 and relays them to the user as required;

d)	maintains call/service state information as perceived by the 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 as "requested" by the CCAF;

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

The SS function (SSF): The SSF is the service switching function, which, associated with the CCF, provides a 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 SCF;

b)	manages signalling 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.

The SC function (SCF): The 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 functional entities to access additional logic or to obtain information (service or user data) required to process a call/service logic instance. It:

a)	interfaces and interacts with service switching function/call control function, 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.

The SD function (SDF): The SDF contains customer and network data for real-time access by the SCF in the execution of a service. It interfaces and interacts with SCFs as required.

NOTE – The SDF contains data relating directly to the provision or operation of services. Thus, it does not necessarily encompass data provided by this party (such as credit information), but may provide access to these data.

The SR function (SRF): The SRF provides the specialized resources required for the execution of services (e.g. digit receivers, announcements, conference bridges, etc.). It:

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

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

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

Call-Related Radio Access Control Function (CRACF): The CRACF supports service/features and signalling that require handling and manipulation of the radio link. It:

a) provides the mechanisms to detect call-related interactions that require functionality that exists within the radio access control function or within other FEs;

b) is managed, updated and/or otherwise administered for its call related functions.

Radio Control Function (RCF): The RCF assists in providing access to users/terminals. It provides the network call-related and call-unrelated radio link functions. It:

a) establishes and maintains, modifies, and releases a radio and fixed bearer line connection between a mobile terminal and the network;

b) access the call-related, radio link capabilities of the CRACF using service requests;

c) receives indications from the user agent/terminal agent and/or determines the state of the radio link connections between the terminal and network and relays from them to the CRACF as required;

d) maintains state of the radio link connections, for a given call, between a user agent/terminal agent and the network as perceived by this functional entity.

2.1.3 Functional entity relationships

Services are supported by the cooperative actions of a set of functional entities. Cooperation requires that communication relationships be established:

- Each communicating pair of functional entities in a specific service functional model is said to be in a relationship.
- Each interaction between a communicating pair of functional entities is termed an information flow. The relationship between any pair of functional entities is the complete set of information flows between them.
- If a communication pair of functional entities is located in physically separate devices, the information flows between them define the information transfer requirements for a signalling protocol between the devices.
- Different communicating pairs of functional entities may have relationships of different types. The type of a relationship is characterized by the set of information flows between two functional entities. The relationships between functional entities FE1 and FE2 and between functional entities FE3 and FE4 are said to be of the same type if they comprise the same set of information flows.

- Relationships are assigned type identifiers (e.g. r1, r2, r3, etc.) which uniquely identify specific sets of information flows within the functional model of a service. The same relationship type may occur more than once in a functional model.

2.1.4 Derivation of the functional model

Based on the above definitions, the functional model for a particular service is derived using the following criteria and guidelines:

– Appropriate functional entities are chosen based on knowledge of the variety of anticipated network realizations. All reasonable distributions of functions should be considered, thus leaving the option open to an Administration as to how actually to offer the service.

– Relationship types are initially assigned based on an assessment of the probable nature of the interactions between each pair of functional entities. Revisions to the initial model may be necessary in the light of more detailed definition of functional entity actions, information flows and the range of physical locations for functional entities.

– The model for some services may require that a functional entity be replicated a number of times (e.g. tandem functions). The functional model should only describe replications up to the point where no new combinations of external relationships to functional entities are encountered by further replication. Thus, a single function entity may represent multiple physical tandem entities providing the same function.

The unified functional model is intended to capture aspects from as many network architectures as possible. For ISDN, although many of the FEs in the model are normally co-located in physical implementations, the service functionality can be separated among the functions specified. For IN and FPLMTS, the functional model and FEs are typical of IN capability set and FPLMTS architectures. For B-ISDN, the unified functional model identifies edge-to-edge relationships between serving call control functions, and link-by-link relationships using transit call control functions. Point-to-multipoint configurations are catered for by identifying additional instances of similar functional entities and relationships as appropriate. Managed objects can be considered to exist within FEs and can, as such, be addressed and accessed through those FEs.

2.1.5 Relationship between basic and supplementary service models

The functional model for a supplementary service shall relate to a basic service model. The following guidelines should be followed in resolving whether the functions associated with supplementary service should be co-located with existing basic service functional entities or in the form of new functional entities that are not co-located:

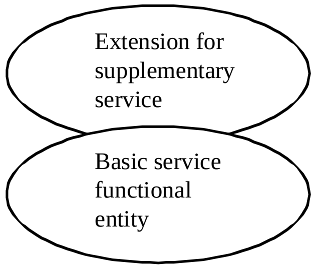
– A grouping of functions within a supplementary service model should be co-located with a basic service functional entity (e.g. see Figure 5) if it modifies an object (e.g. call connection) that is controlled by the basic service.

– A functional entity that is not co-located with a basic service functional entity typically would not require detailed call/connection state information. A separate functional entity may also be characterized by having a transactional relationship with a functional entity of the supplementary service co-located with a functional entity of the basic service (e.g. to provide number translation to the basic service functional entity).

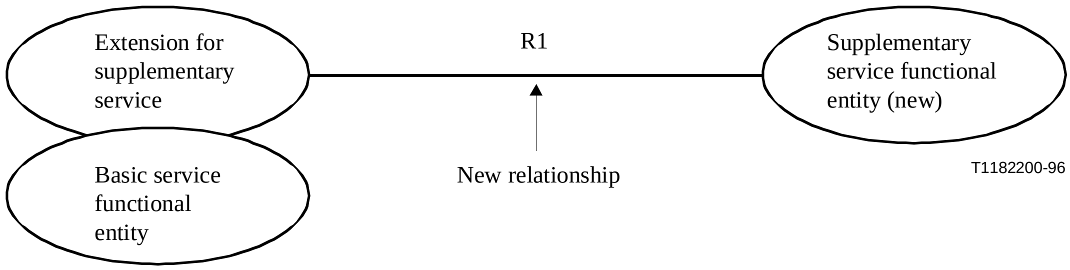
– A relationship between the functional entities for the supplementary service is defined if, and only if, the supplementary service functional entities are required to communicate and those communication needs are not met by the information flows defined within the basic service.

– The relationship between the model for a supplementary service and that for a basic service may be derived by comparing the models. How the functional entities of the supplementary service needs to take into account all scenarios defined in Step 6 for both the basic service model and for the supplementary service model.

Figure 4 illustrates these relationships.



a) Additional functions for supplementary service provided as an extension to a functional entity of a basic service



b) Additional functions for a supplementary service provided as a separate non-located functional entity

Figure 4/Q.65 – Alternative ways of adding supplementary service functions to the basic service functional model

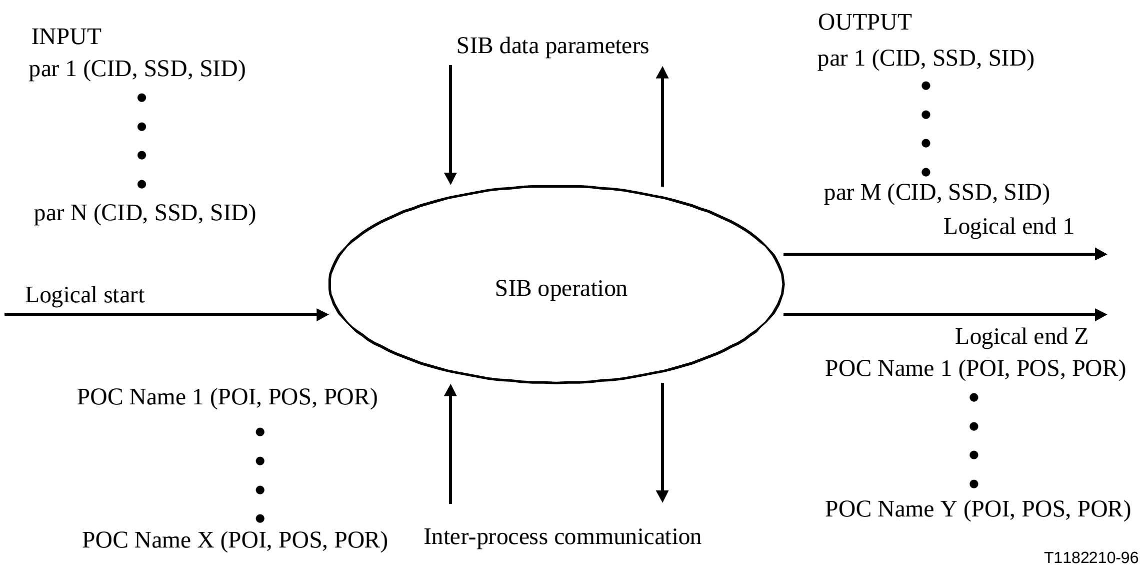


Figure 5/Q.65 – SIB operation graphical representation

2.2 Step 2 (Optional) – SIB description of service features

Service independent building blocks

A SIB is a standard reusable network-wide capability residing in the global functional plane used to create service features (see Figure 1). 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 (DFP) and the physical plane. The SIBs are reusable and can be combined to realize services described in the service plane. The capability offered within the SIB is described by the set of operations that may be invoked in the SIBs. The set of operations that the SIB offers constitute the SIB interface. Each operation defines a function that can be performed related to the SIB capability. Complex SIBs, e.g. those modelling persistent activities are defined on the basis of several operations that allow the control of the activity performed by the SIB. SIBs are defined to be independent of the specific service and technology for which or on which they will be realized.

Characteristics of a SIB

- 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 combination 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 functional entity actions which may reside in one or more functional entities (FEs).
- A SIB operation has one logical starting point and one or more logical end points. Data required by each SIB operation is defined by SIB operation support data parameters and call instance data parameters.
- SIBs are global in nature and their location 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.

This subclause is intended as an introduction to the use of Service-Independent Building Blocks (SIBs) in the Unified Functional Methodology (UFM) for service descriptions. The concept of SIBs has been adopted to address service creation needs as well as to introduce re-usable blocks of flows, SDL, and FEAs which can be catalogued. The addition of the SIB concept is introduced to streamline the work of Stage 2 service definition. New SIBs will be created as needed, and a SIB library will be maintained as a reference for service creation and definition.

NOTE – The use of SIBs for Stage 2 descriptions is not confined to IN-supported services only.

The IN conceptual model approach of defining and using SIBs provides the basis for a flexible service description and creation method, as well as a method for referencing pre-defined flows and SDL. The SIB concept [as referenced in Recommendations Q.1213/Q.1214 (1995)] has been chosen as a key addition to the Stage 2 methodology described in this Recommendation.

Common ground for the unified methodology exists in that SIBs are decomposed into FEAs, SDLs, and information flows. The appeal of the method is in the potential for re-use (SIBs map to predefined functional entity actions described by pre-defined information flows and SDL). It cannot be absolutely proven that SIBs may be combined in all possible ways to create new services, but several methods are being considered (e.g. extension of the SDL process to include the combination of SIBs). Verification of the existing SIBs (i.e. that each is correctly defined) is possible by analysing the SDLs which define them. Verification of SIB combinations is possible by analysing the SDLs created when SIBs are combined to describe service features and network capabilities.

There is a need for new SIBs as well as the development of combination rules and application rules. SIBs must be created "top down" from target services, but also "bottom up" from experience with service description methodology. In this way, the methodology can address current needs, but also evolve to include improvements and new technologies. With continued efforts on SIB definition and application rules, SIBs could become the primary tool for Stage 2 service description and service creation.

2.2.1 SIB definitions

Terminology:

BCP	Basic Call Process
CID	Call Instance Data
GSL	Global Service Logic
HLSIB	High Level SIB
POC	Point of Control
POI	Point of Initiation
POR	Point of Return
POS	Point of Synchronization
SID	Service Instance Data
SSD	Service Support Data

basic call process SIB

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 service process, which provides event processing capabilities of the basic call, as well as a specialized SIB, which provides a set of SIB operations, such as:

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

high level SIB (HLSIB)

High level SIBs (HLSIBs) are, as normal SIBs, a reusable part of a service feature, and are composed out of SIB operations and other HLSIBs which can be executed sequentially. HLSIBs have the following additional characteristics:

- HLSIBs can be composed out of other HLSIBs and SIB operations only.
- A certain HLSIB can not be used as a component within the same HLSIB, i.e. no recursive use is possible.
- The lowest level of HLSIBs contains SIB operations only, i.e. no further detail is visible on the GFP.
- One of the (HL)SIBs within a HLSIB is the first to be executed; therefore, HLSIBs have only one entry point (logical start), the same as with normal SIBs. But, as with normal SIBs as well, HLSIBs can have one or more exit points (logical ends).

global service logic

The GSL can be defined as the "glue" that defines the order in which SIB operations will be chained together to build service processes to accomplish service features. Each instance of global service logic is (potentially) unique to each individual call, but uses common elements, comprising specifically:

- interaction points (POI, POS and POR) of service processes, including the BCP Service Process;
- SIB operations;
- logical connections between SIB operations, and between SIB operations and service process 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.

2.2.2 SIB data parameters

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

In order to describe service features with these generic SIBs, some elements of service dependence are needed.

Service dependence can be described using data parameters which enable 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.

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

Differentiation of formal SIB parameters from actual parameters introduces more flexibility in assigning the SIB parameter data type. The formal SIB parameters are the parameters that will be used for SIB descriptions in this Recommendation. Actual SIB parameters only occur in SIB instances in specific Global Service Logic (GSL).

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 (e.g. Calling Line Identification);
- generated by a SIB operation, (e.g. a translated number); or
- entered by the subscriber, (e.g. a dialled number or a PIN code).

service support data (SSD)

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

service instance data (SID)

Service instance data defines data related to a service subscriber's profile, that exists before the service is invoked and can be modified and updated as a result of the service processing activity. This type of data can be read within the service execution and be stored to be used in further service invocations.

2.2.3 SIB modelling conventions

Graphical representation

A graphical representation is used to describe each operation performed by a SIB. It is illustrated in Figure 6. Each SIB operation is characterized by having input and output parameters, one input logical flow, and one or more output logical flows. 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. Input and output parameters are identified by the dashed arrows at the top of the diagram and are specified beside the dashed arrow. For both input and output parameters, SSD, SID and CID type is declared beside the respective parameters. Similarly, POCs are specified below the diagram.

SIBs can be defined with various degrees of granularity. By composition, SIBs can be defined out of smaller SIBs, forming a High Level SIB (HLSIB). Conversely, decomposition allows the partitioning of the granularity of a HLSIB into smaller blocks that can be re-used. Figure 6 shows several layers of granularity for HLSIBs. It may be advantageous to catalogue certain HLSIBs as well as simple SIB combinations in order to promote re-use of more complex functionality.

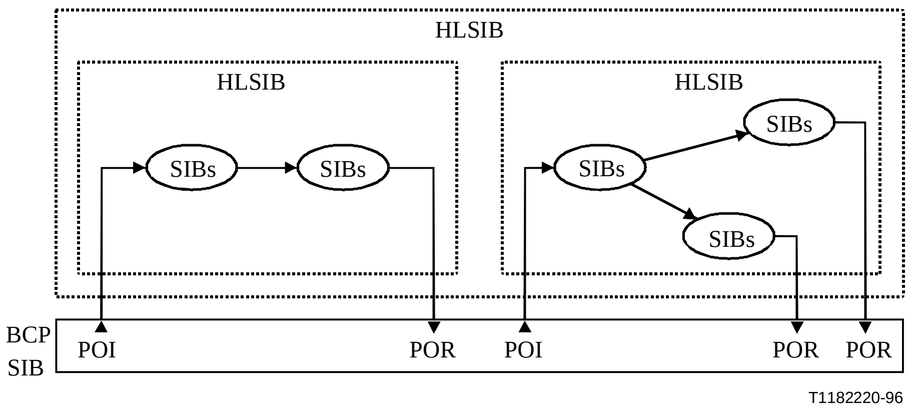


Figure 6/Q.65 – SIB combination/HLSIBs (generic example)

Figure 7 shows an example of a global service logic diagram. The global service logic diagram, which shows the SIB, or SIBs in combination, performing a particular service function is a perfect "bridge" from a Stage 1 description to the Stage 2 description. The diagram describes the service or network capability in a shorthand way that conveys a great deal of information with a relatively simple graphical means. The definitions contained in the diagram, however, contain all the SDL, information flows, and FEAs necessary to complete the Stage 2 description.

A possible definition of the terminating screening service is shown in Figure 7. From the cell arrival POI, the SCREEN SIB is used to determine if the calling user is on the list of users allowed to terminate a call at the destination. If on the list, the call is permitted, and the BCP continues call handling with existing data. If the caller user is not on the list, the USER INTERACTION SIB is used to deliver an appropriate disconnection message to the caller, at which time the BCP clears the call.

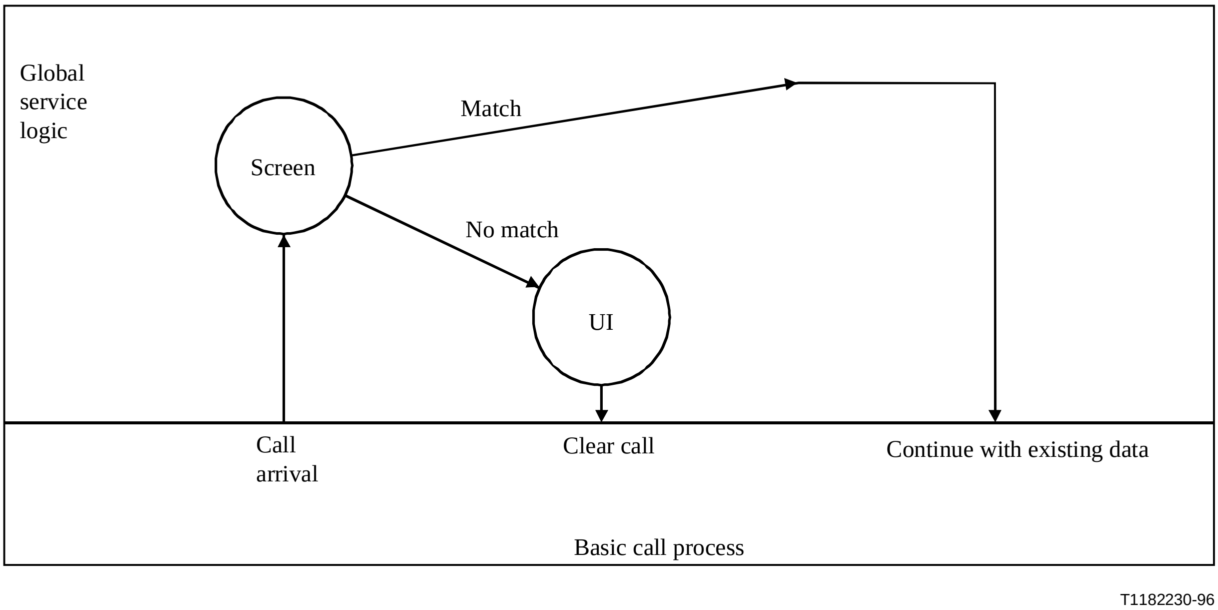


Figure 7/Q.65 – GFP terminating screening service

2.2.4 SIB modelling of service features

It is encouraged that SIBs and HLSIBs be used to describe service features and network capabilities, since it promotes the re-use of SDL, information flows, and FEAs which are already defined. The global service logic diagram, showing the SIB, or SIBs in combination, performing a particular service function is an efficient and effective means of providing the SIB input to the Stage 2 description.

The following are guidelines for the use of SIBs in the Stage 2 method:

- Provide GSL diagrams of service features including POI(s) from and POR(s) to the BCP SIB.
- Provide a mapping of the SIBs used in the description to the FEs of the unified functional model. Also, provide references for the definitions of the SIBs used [e.g. clause 5/Q.1213, (1995)]. This can easily be done in tabular form (see 2.2.6).
- Provide SIB data parameters needed for the service. Identify data values within the context of the service.

If all service features can be described completely with SIBs, specific references should be provided for the supporting SDL, information flows, and FEAs in the appropriate sections of the Stage 2 description. Explicit drafting of these sections is not needed (provided the SIB descriptions have been proven to be accurate. These "proofs" for various SIB combinations and HLSIBs will come with experience; the SIB library will preserve this information). If there are no SIBs available to describe a particular feature, the unified functional methodology can still be used as it has always been used to explicitly derive the SDL, information flows, and FEAs. If some SIBs are identified to partially describe a service, it is suggested that the SDL diagrams and information flows be explicitly shown with indications as to which sections of flows are from SIB definitions.

2.2.5 List of available SIBs

IN CS-1R Service-Independent Building Blocks (SIBs):

1) Algorithm;
2) Authenticate;
3) Charge;
4) Compare;
5) Distribution;
6) Limit;
7) Log call information;
8) Queue;
9) Screen;
10) Service data management;
11) Status notification;
12) Translate;
13) User interaction.

Additional SIBs are defined in the Q.12x3-Series of Recommendations.

2.2.6 Mapping SIBs to FEs

Table 1/Q.65 – SIB/FE mapping

SIB	Functional entities			
	SSF/CCF	SCF	SRF	SDF
Algorithm		X		
Charge	X	X		
Compare		X		
Distribution		X		
Limit	X	X		
Log call information	X	X		X
Queue	X	X	X	
Screen		X		X
Service data management		X		
Status notification	X	X		
Translate		X		X
User interaction	X	X	X	
Verify		X		
Basic call process	X	X		
Authenticate		X		X

2.3 Step 3 – Information flow diagrams

2.3.1 Identification of information flows

The distribution of the functions required to provide a service, as defined by the functional model, requires that interactions occur between functional entities. Such an interaction is referred to as an "information flow" and will have a name descriptive of the intent of the information flow.

Information flow diagrams are created to contain all the information flows necessary for typical cases of successful operation of the service. Information flow diagrams may need to be created as appropriate for other cases. Figure 8 illustrates the general form of an information flow diagram for a basic or supplementary service.

Information flow diagrams for supplementary services should not necessarily duplicate information flow descriptions that are part of a basic service. However, it may be that a supplementary service description identifies additional information flow requirements between the functional entities of the basic service representation, and this should be described.

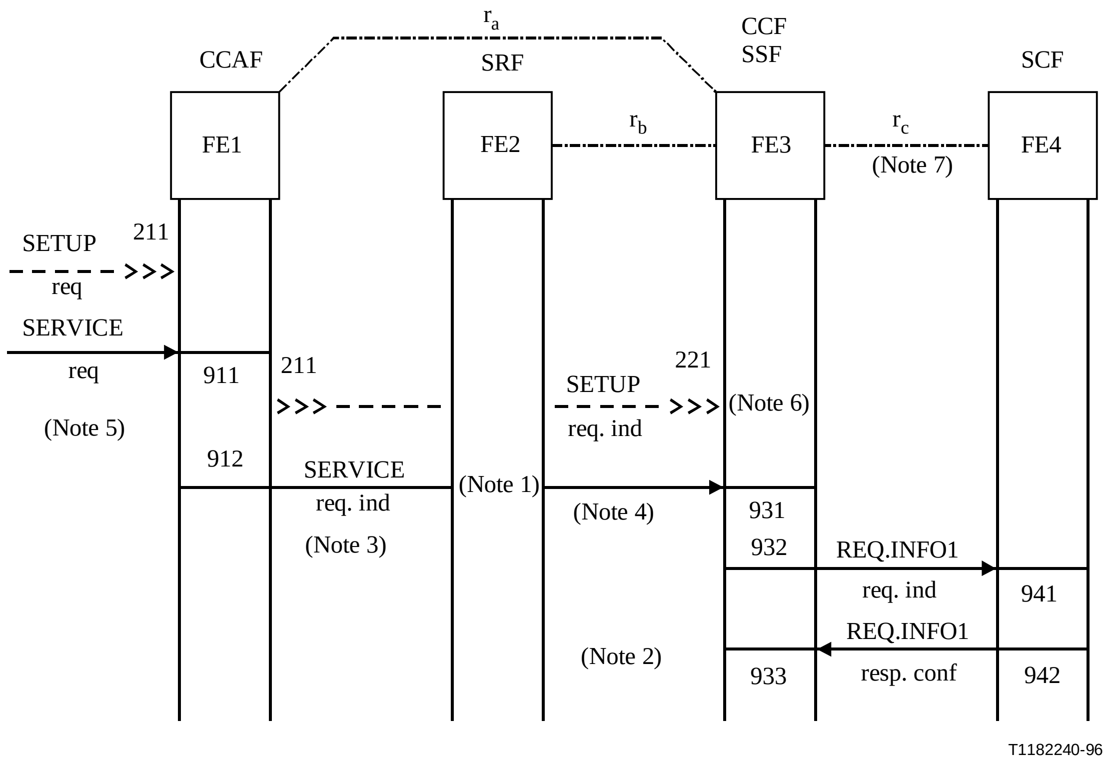


Figure 8/Q.65 – Example of information flow diagram

Notes to Figure 8:

NOTE 1 – Receipt and emission of user inputs/outputs and information flows are shown by horizontal lines across the relevant functional entity columns. Conversely, the absence of a line indicates no receipt or emission.

NOTE 2 – A reference number is assigned to each point in the overall sequence at which functional entity actions are shown.

NOTE 3 – Information flows are shown as arrows with the name of the information flow above and below the arrow. The descriptive name is written in capitals above the arrow and the label (e.g. req. ind.) is written below line in lower case. For unconfirmed information flows and the "request" part of confirmed information flows the label "req. ind." is shown in lower case below the information flow arrows. For the "confirmation" part of confirmed information flows the "resp. conf" is used.

NOTE 4 – In a particular functional entity column:

– Actions shown below a line representing the receipt of a user input or information flow are dependent upon that receipt (i.e. they cannot be carried out beforehand). Thus Action 931, for example, cannot be carried out before SERVICE is received.

– Similarly, actions shown above a line representing the emission of a user output or an information flow must be completed prior to the emission flow. Thus, REQ.INFO1 cannot be emitted until Actions 931 and 932 are both completed. No implications regarding the order of execution of Actions 931 and 932 are intended.

– Actions shown below a line representing the emission of user output or information flow do not need to be completed before emission (although in many practical implementations they may). No constraint on the relative order of the emission and the action which immediately follows it is intended. Thus Action 942 may be executed before, after or in parallel with emission of the "request" part of the REQ.INFO1 information flow.

NOTE 5 – The Stage 1 service interactions are inputs to and outputs from the Stage 2 information flow diagram. Stage 1 service interactions from the user are either of the form XXXXX.req or XXXXX.resp. Stage 1 service interactions to the user are either of the form xxxxx.ind or XXXXX.conf.

The following guidelines are observed in drafting these information flows diagrams:

- Vertical columns represent each of the functional entities identified in the functional model for the service. Information flows are shown in descending order in which they occur in the processing of a call. The order of functional entity actions shown between information flows is not significant.

- An information flow will be characterized in the arrow diagrams as being associated with the terms request/indication or response/confirmation. This is reflected in the primitive which is communicated to the underlying signalling system as illustrated in Figure 8. The primitive name is, in general, a direct derivation of the information flow name. The terms are shown in association with the information flow to show the relation between the Stage 2 SDL and the SDL of the underlying signalling system.

A reference number uniquely identifies a particular point in the Stage 2 information flow sequence and appears on the information flow diagram at that point. It also serves as a pointer to a description (see 2.4 below) of the actions required at this point in the sequence. A brief description of the functional entity actions may also appear on the relevant part of the information flow diagrams. The reference numbering scheme to be used is described below.

Each number is of the form XYZ and is a number assigned by the drafter of the Stage 2 description, which identifies a particular point in Stage 2 procedural description (arrow diagrams and SDL) at which functional entity actions are described. For supplementary service actions, X of the FEA number must be "9". Y is the number of the FE where the action is executed. Z enumerates the actions in a single FE (Z=1,...,9,A,...,Z,a,...,z). This number is unique within the Stage 2 description of a particular service (all variants).

NOTE 6 – Basic Call flows are shown by dashed lines and chevrons. Supplementary service flows are shown by solid lines and arrow heads. Information transferred "in-band" is shown by double dashed lines.

NOTE 7 – The relationships between FEs may also be shown on the diagram.

2.3.2 Definition of individual information flows

The semantic meaning and information content of each information flow is determined. An individual information flow may be identified as requiring confirmation, and if so, it requires a return information flow of the same name.

Confirmed information flows take the form of a request for an action (in one direction) and confirmation that the action has been carried out (in the return direction). Confirmed information flows are typically required for synchronization purposes. The two main cases are when requesting allocation and/or release of a shared resource.

When interacting functional entities are implemented in physically separate locations, information flows will normally be conveyed by signalling system protocols. When interacting functional entities are implemented in the same location, information flows are internal and do not effect signalling systems protocol.

Tables should be constructed to show all items in each information flow. It should be indicated whether or not each item is mandatory or optional, and along what relationship the information is passed.

Table 2/Q.65 – Example of definition of individual information flows

REQ.INFO1

Relationship	Item	req.ind	resp.conf
r_a, r_b, r_c	Request for billed number	Mandatory	--------
r_a, r_b, r_c	Billed number	--------	Mandatory
r_a, r_b, r_c	Request for service type	Optional	--------
r_a, r_b, r_c	Service type	--------	Optional

2.4 Step 4 – Functional entity actions

The Stage 2 actions performed within a functional entity, from the reception of each information flow to the transmission of the next resulting information flow, are identified and listed. All externally visible actions (those which are explicitly or implicitly notified to other functional entities) are included. The identified actions are then represented on the information flow diagrams and SDL diagrams by brief prose statements, or separately using reference numbers.

NOTE – The implementation of the charging principles of the D-Series Recommendations is a national matter. Therefore, FEAs which are solely for charging should be included only if there is a direct need for a signalling support, or if the service may be provided by public networks across an international boundary. In cases where the charging principles of the D-Series Recommendations need to be reflected in Stage 2 service descriptions, the following footnote should be entered for each FEA that contains charging.

"This FEA describes actions that might be adopted by Administrations to implement the charging principles of the D-Series Recommendation."

This functional entity actions subclause contains descriptions of actions required for each functional entity and each FEA is identified by a reference number.

The presentation form for functional entity actions is illustrated in Figure 9.

FEAs of FE2

921:

- Interact with user to accumulate information
- Select network access resource
- Reserve facilities, both directions as required

922:

- Interact with user to obtain call address
- Determine and indicate end of dialing

Figure 9/Q.65 – Example of descriptions of functional entity actions

2.5 Step 5 (optional) – SDL diagrams for functional entities

The system description and specification language (SDL, see Recommendation. Z.100) in its graphic representation is used to describe a given service formally in terms of the actions performed in the functional entities involved in providing the service and the information flows which result from or trigger these actions. The SDL diagrams are based on (and are consistent with) the information flow diagrams generated in step 2 of the service description method and describe normal, unsuccessful and abnormal service operation in detail.

The information flows, which are fully defined in terms of their contents and of their origination and destination points as part of the SDL diagram, are supported by signalling procedures specified in Stage 3 service descriptions.

The description of SDL is contained in Recommendation Z.100. A summary of its elements is contained in the following section.

NOTE – Step 5 (SDL) is considered optional for Stage 2 only if formal SDL diagrams are specified in the Stage 3 description.

2.5.1 SDL general aspects

The elements of a service description in SDL are systems, blocks, processes and procedures (see 2.4/Z.100 and Annex B/Z.100). Systems, blocks and processes can be defined as an instantiation of a system type, block type or process type. A system (type), block (type), process (type) or procedure is described in one or more diagrams. Procedure diagrams are present if process diagrams include procedure calls.

Information between blocks or between processes is conveyed in signals which may have parameters containing data. Data is defined in formal data type definitions.

2.5.1.1 The system diagram

In the context of the Stage 2 description of a given telecommunication service the system type diagram describes the system in terms of blocks, channels and signals. Blocks represent the set of network and network access functions that need to be performed by the system in order to provide a service. Signals conveyed over channels (see 2.5/Z.100) describe the information exchanged between blocks and between blocks and the system's environment. Channels are connected to blocks or the enclosing frame of a diagram via named *gates* if the block is defined as an instance of a block type. The list of signals carried on the channels of the system is part of the system diagram.

An example the system diagram for a system named BasicService (assumed to be the circuit switched bearer service) is shown in Figures 10a and 10b. The diagram in this case encompasses two frames named "Definition" (Figure 10a) and "Structure" (Figure 10b).

NOTE – The number of frames used to draw a diagram may be chosen arbitrarily and is usually based on available drafting space and legibility considerations.

The frame designated "Definition" contains:

a) a block type symbol which contains the name of the block type (BS_St2_BT) preceded by the keyword virtual. This signifies that block type BS_St2_BT can be redefined, e.g. when it becomes necessary to incorporate additional service logic in the basic service to support a supplementary service;

b) a text symbol containing the list, headed by the keyword SIGNAL, of the signals that can be transferred between the system and the environment;

c) text symbols containing descriptions of lists of signals (keyword SIGNALLIST) named Afw, Abw, Bfw and Bbw, respectively;

d) a text symbol containing the definition (keywords NEWTYPE, ENDNEWTYPE) of a data type RejectCause. Variables of this type can assume the values (keyword LITERALS) RequestNotValid, NoRoute, NoBresources, NoResponse and Busy. Standard data types [e.g. Boolean, Integer (see Annex D/Z.100)] need not be defined again in service descriptions;

e) a text symbol containing a list of synonyms which are not specified in the service description. The keyword SYNONYM indicates that names are given to externally defined values (keyword EXTERNAL), e.g. NoAnswerTime which is of standard data type Duration stands for a value which is defined externally.

The frame designated "Structure" describes the blocks and channels of system BasicService. In this case, the system contains an instance BS_St2 of block type BS_St2_BT which encompasses all basic service-related network actions. The users of the service are located outside the system, i.e. in the environment. Their actions are not described in Stage 2, only their interactions with the network. These are represented by the signals defined in the signal lists and are transferred over named channels to and from the environment. Thus, the signals in signal lists AFw and ABw are transferred to and from the environment over the channel called UserAInterface which is connected to block instance BS_St2 at gate To_A. Similarly, the signals in signal lists BFw and BBw are transferred to and from the environment over the channel called UserBInterface which is connected to block instance BS_St2 at gate To_B.

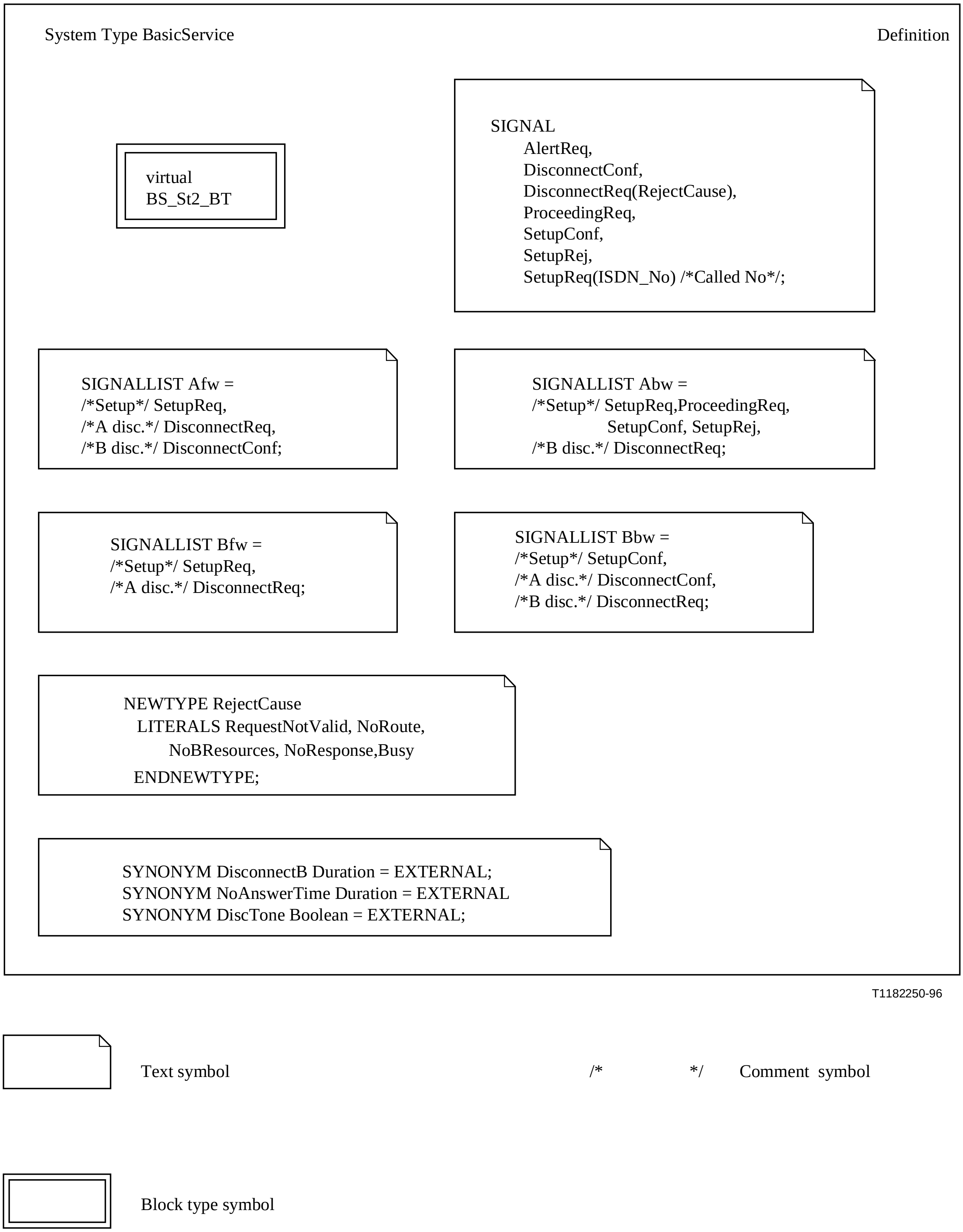


Figure 10a/Q.65 – Example of a system type definition diagram

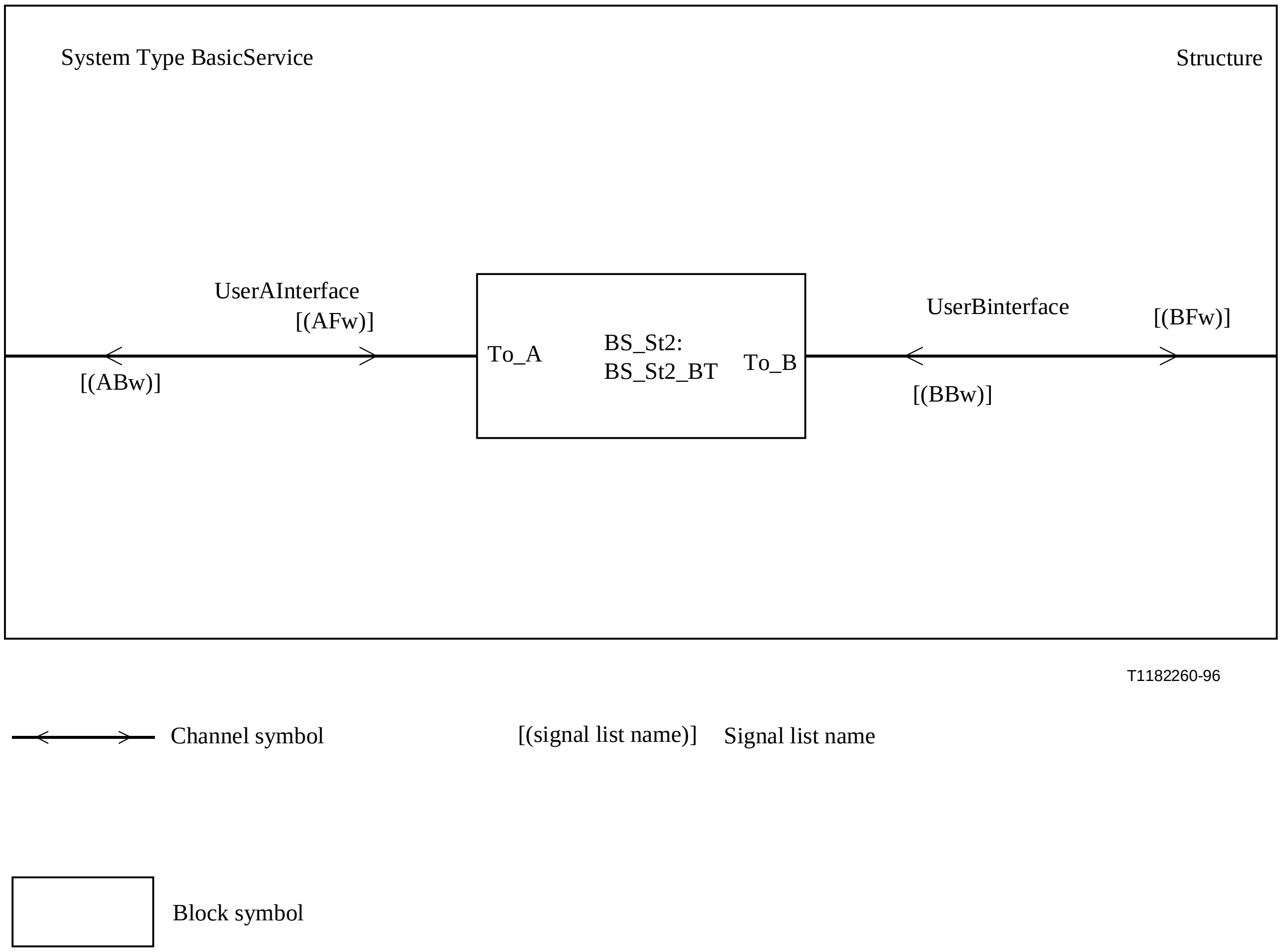


Figure 10b/Q.65 – Example of a system type structure diagram

2.5.1.2 The block diagram

In the context of a Stage 2 service description, the block type diagram defines the process types used in the system and the way in which instances of these types are structured and communicate in order to support a service. The frames of Figures 11a (Definition) and 11b (Structure) represent the diagram for block type BS_St2. As already noted in the system type diagram, this block type has been defined as a virtual type to allow redefinition if required. The block encompasses instances of five process types, located in the five functional entities (i.e. the originating and terminating call control agent functions, and the call control originating, transit and terminating functions) which are involved in initiating and terminating an instance of the basic service or, in other words, setting up and releasing a basic call.

For the basic service example, the five process types have been defined as virtual types, to allow future redefinition (e.g. to insert supplementary service logic).

In the process diagram, information transfers occur over signal routes. These either connect process instances to other process instances or process instances to channels through named gates. For example, route R1 interconnects process instances CCAFO and CCFO via gate OG in CCAFO and gate IC in CCFO and route UIA connects process instance CCAFO to channel UserAInterface via gate To_A. The signals transferred across each route are described in signal lists.

Process symbols contain the names of the process instance and type separated by a colon and two numbers, e.g. (1,1), following the process instance name. These represent the number of instances of the process which exist when the system is created and the maximum number of simultaneous instances of the process, respectively.

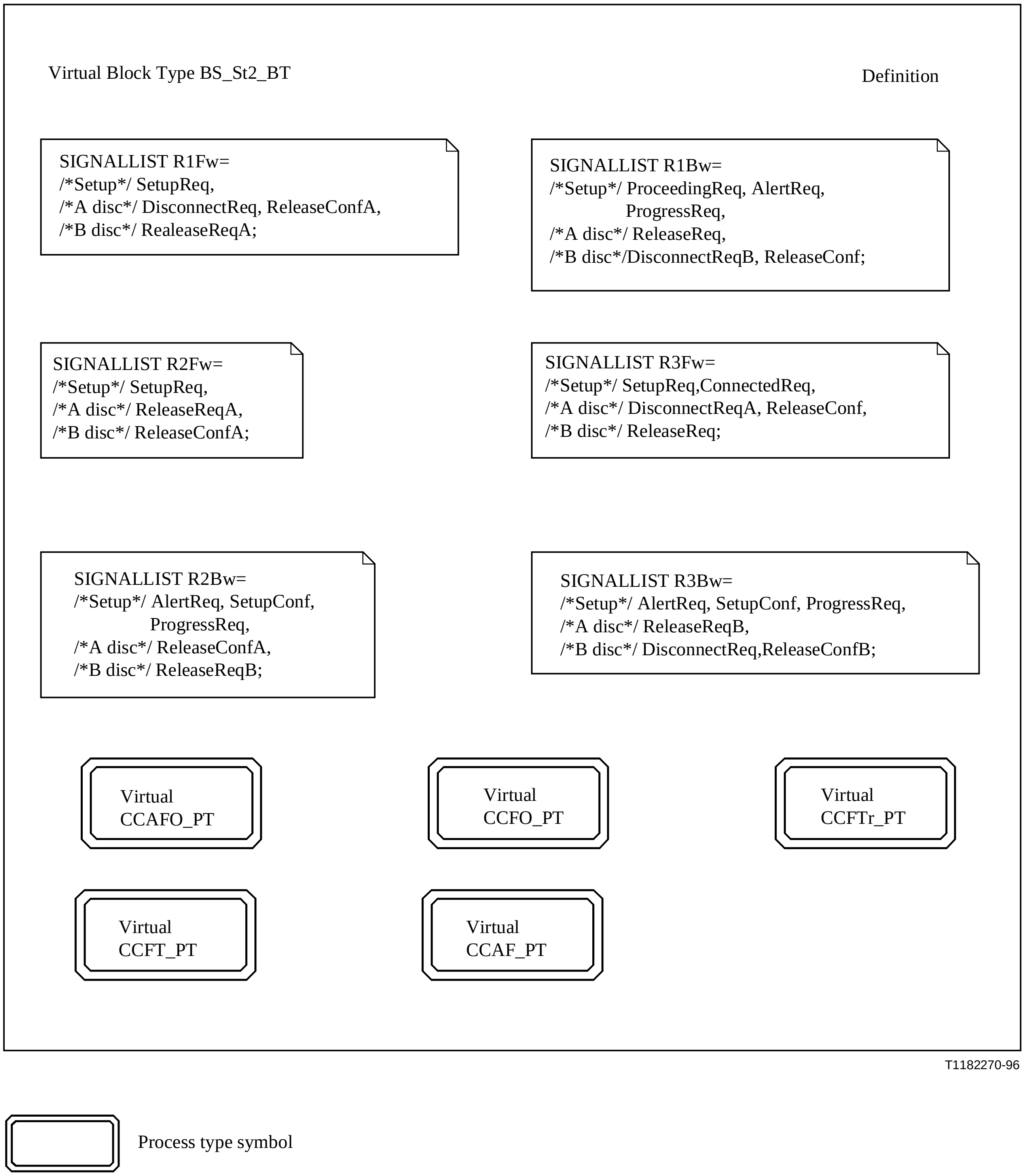


Figure 11a/Q.65 – Example of a block type definition diagram

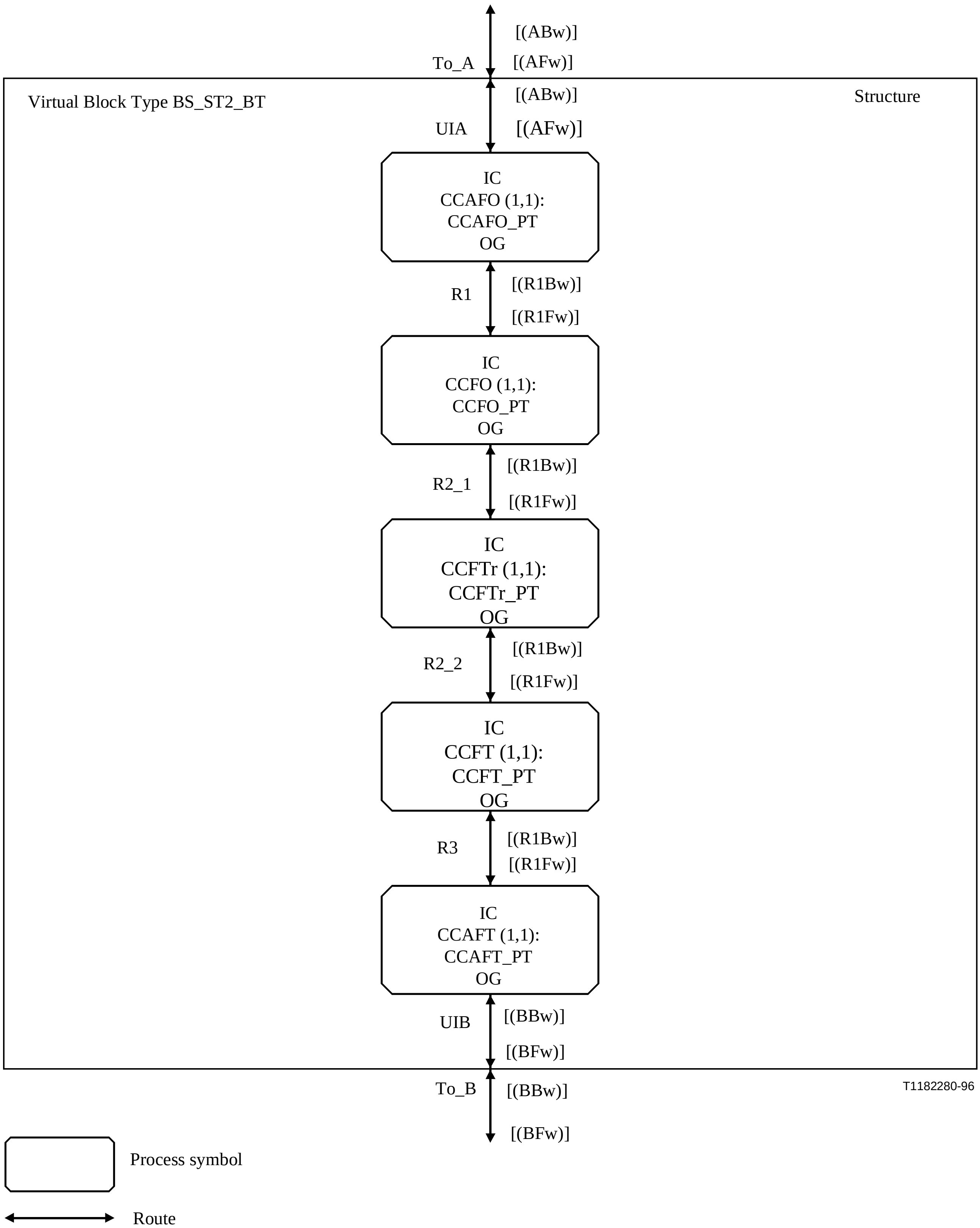
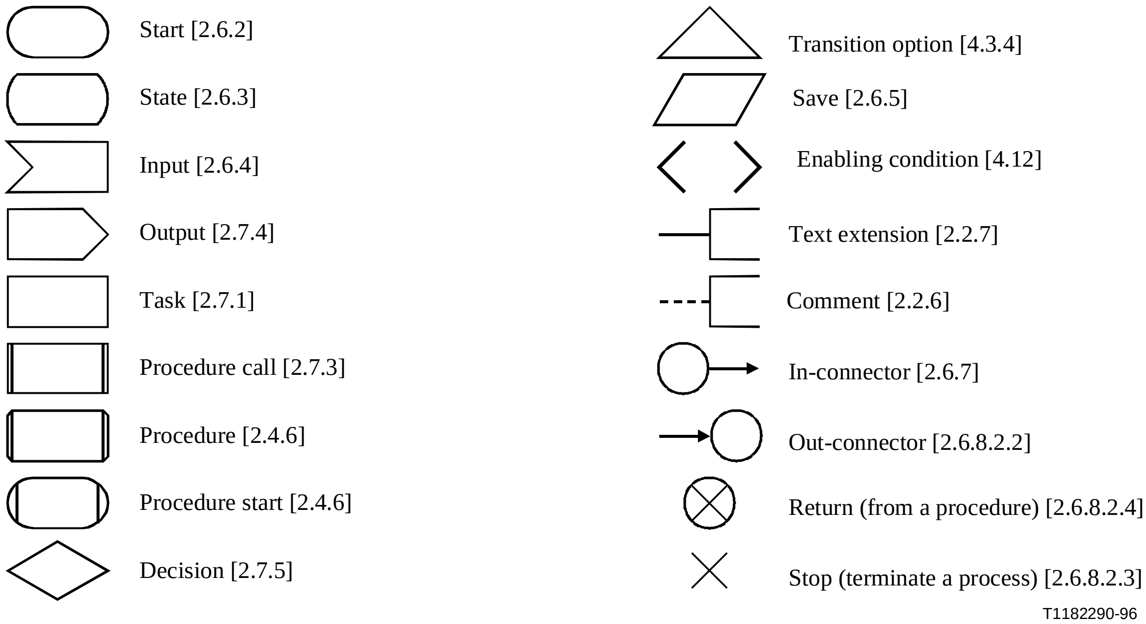


Figure 11b/Q.65 – Example of a block type structure diagram

2.5.1.3 The process diagram

A process diagram describes the sequence of actions (transactions) which are executed by a process instance when a signal is received. Table 3 shows the graphical symbols which are most frequently used in process diagrams of service descriptions. Numbers in square brackets indicate the subclauses of Recommendation Z.100 in which these symbols are described further.

Table 3/Q.65 – SDL symbols



Two examples of process diagrams are shown. Figure 12a contains a fragment of virtual process type CCAFO_PT of block type BS_St2_BT and Figure 12b a fragment of virtual process type CCFT_PT (see Figures 11a and 11b).

Referring to Figure 12, the following explanatory notes apply:

a) The text symbol contains the declaration (keyword DCL) of the variables used by the process. For each variable a name and data type is defined, e.g. the variables named calling and called are of the predefined type PId (process instance identifier) and thus are used to identify a process. Also of a predefined type (i.e. Boolean) are the variables Connected and Resourcesoccupied, whereas the variables CalledNr and Reason are of the new types ISDN_No and RejectCause, respectively, which are specific to this system description.

b) Input and output symbols contain the name of the SDL signal that they represent. Bracketed expression(s) following the signal name, e.g. SetupReq (CalledNr) indicate information (a parameter) carried by the signal.

c) Sender, as e.g. in the task symbol Calling:=Sender, is a keyword used to identify the sending process of the signal that activated the current transition. In the example of Figure 13 two processes, Calling and Called, are identified in this way. In this particular case, these processes are in the environment and are not described in the example system description.

G5

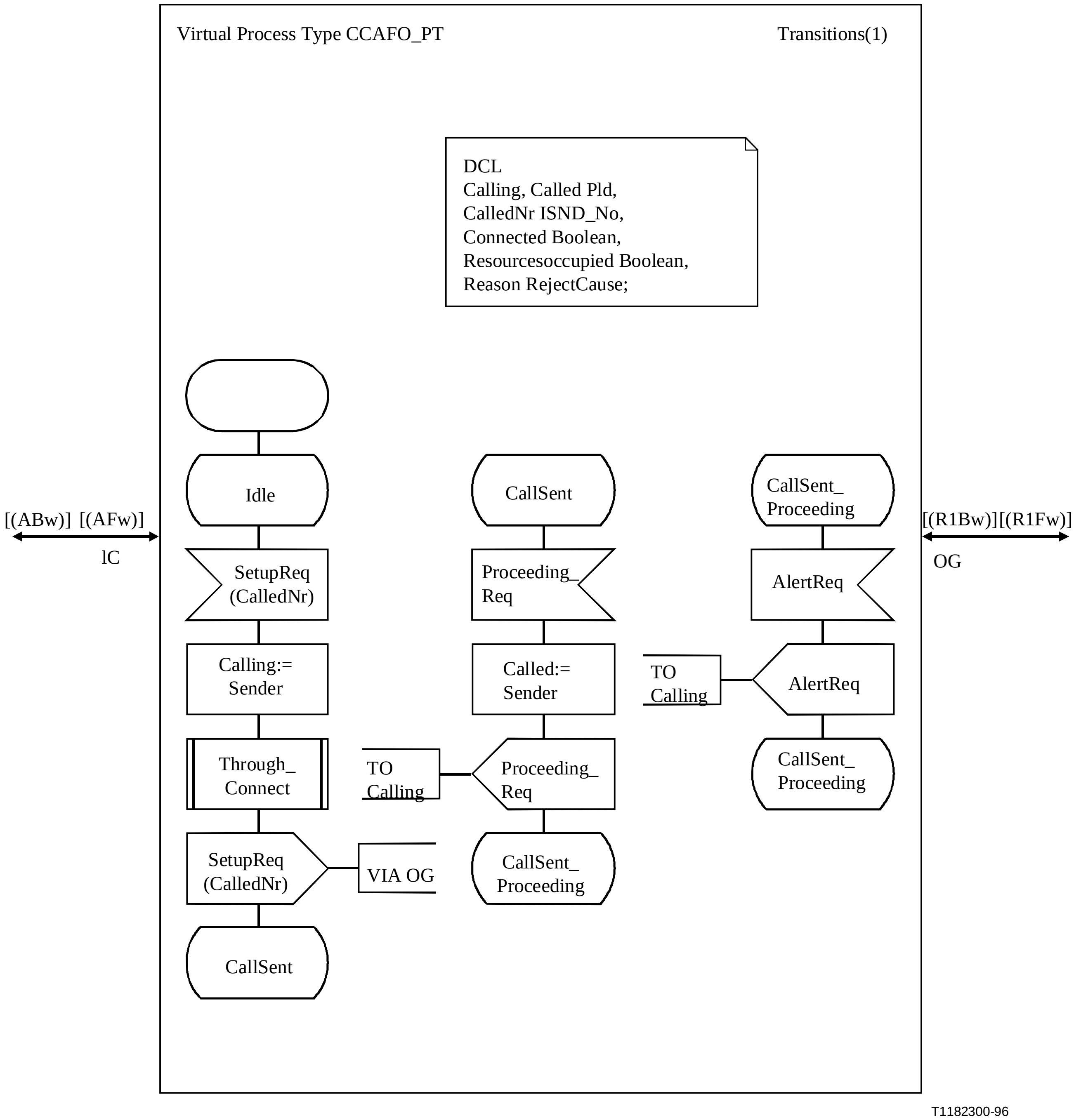


Figure 12a/Q.65 – Example 1 of a process diagram – Fragment of the process CCAFO

Figure 12b illustrates the method of creating an opening for future enhancements in an existing transition. The opening is provided by inserting a procedure call, e.g. to procedure SupSvc1 in the example shown, at a point in the transition where the later addition of logic is anticipated. Initially the called procedure is "empty". It is redefined to provide additional logic whenever that becomes necessary (see 2.5.2) and for this reason is defined as virtual.

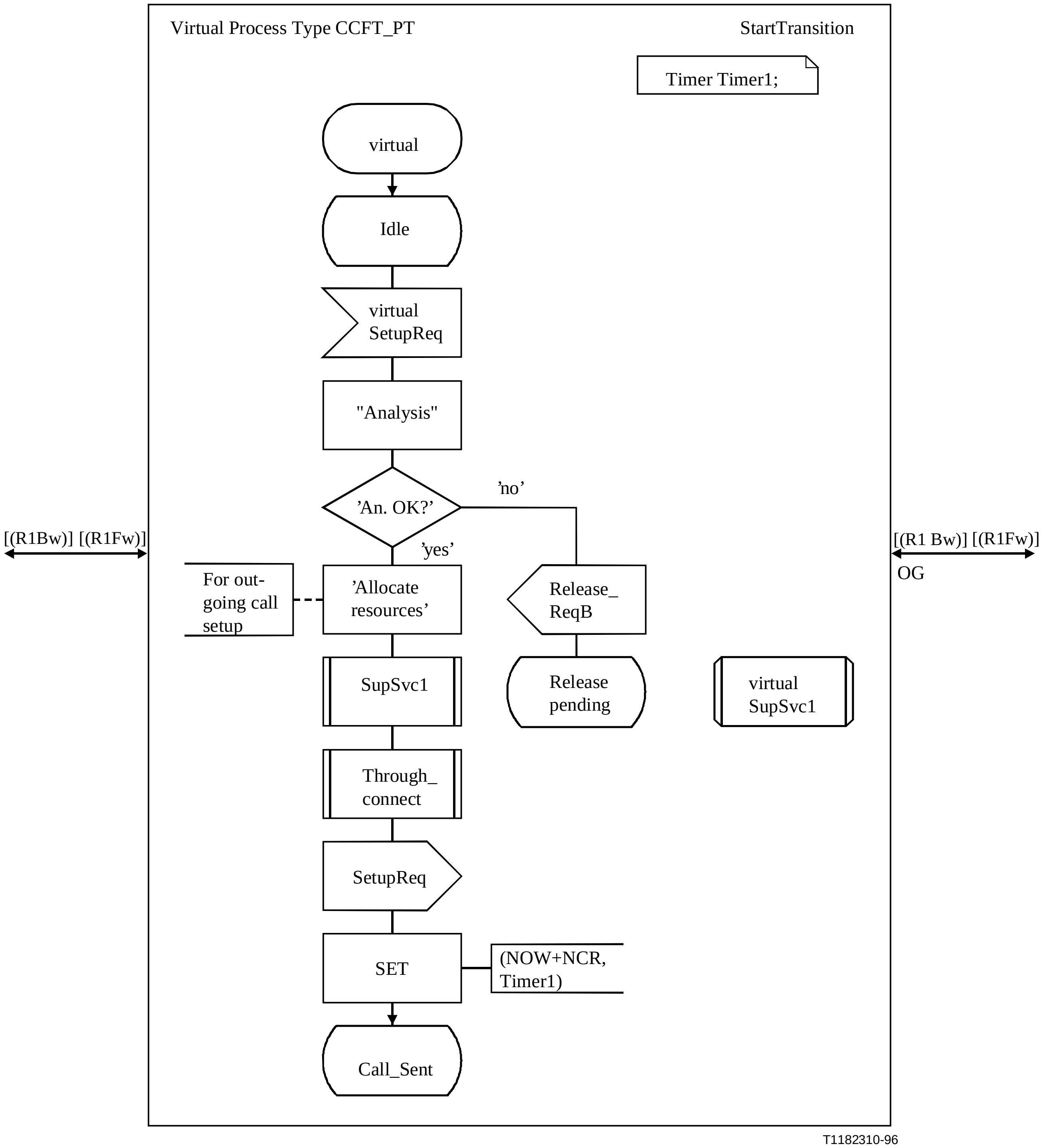


Figure 12b/Q.65 – Example 2 of a process diagram – Fragment of process type CCFT

2.5.1.4 The procedure diagram

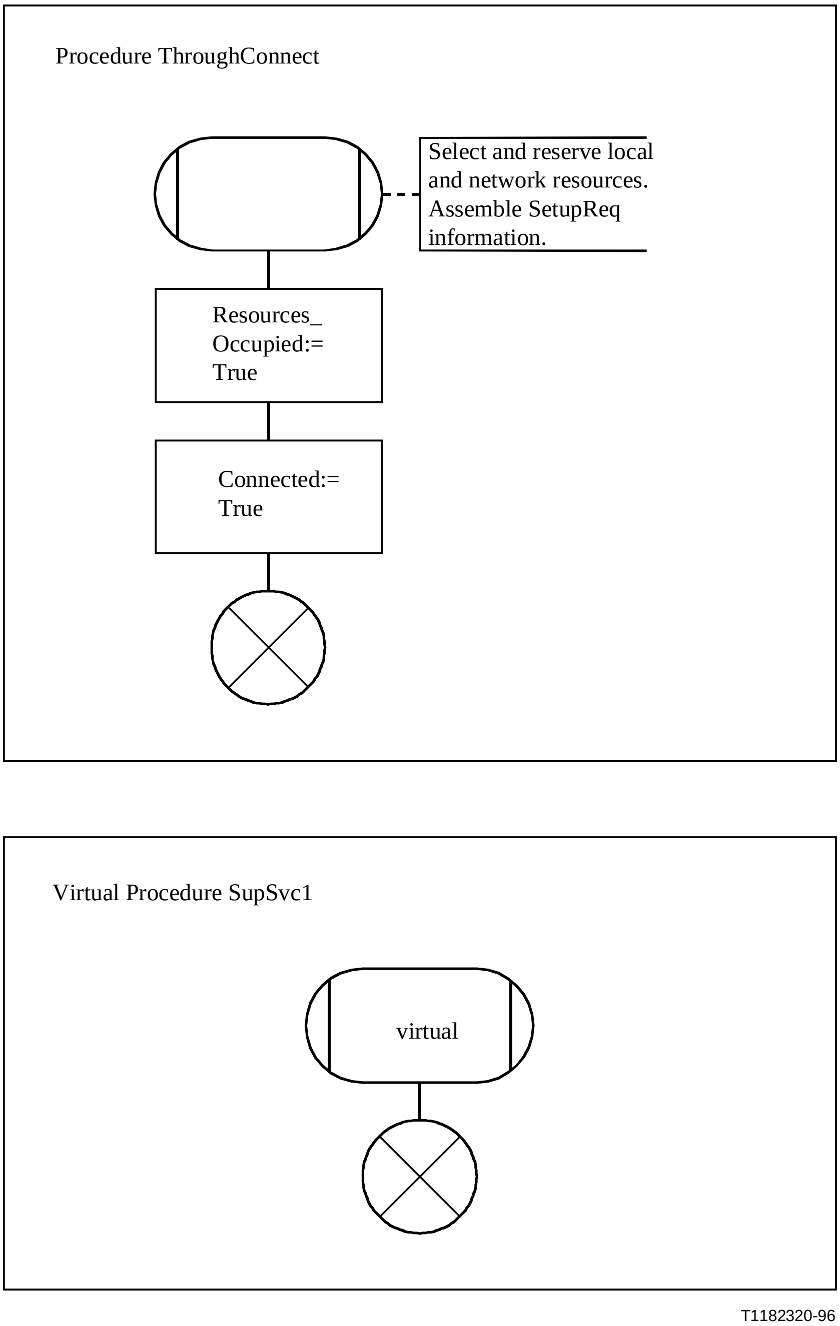


Figure 13/Q.65 – Examples of a procedure diagram

If procedures are called by process types of the system, then these procedures are described in procedure diagrams. The examples in Figure 13 show the procedure ThroughConnect called by process instances CCAFO and CCFT and the procedure SupSvc1 called by process instance CCFT. Since it is anticipated that the latter will undergo modification, it has been defined as a virtual procedure. As noted before, the procedure SupSvc1 is empty and acts for the time being simply as placeholder for the addition of service logic in the future.

The symbols used in the diagrams are explained in Table 3

2.5.2 Adding functionality to existing Stage 2 service descriptions

2.5.2.1 Specialization

Occasionally the need arises for the addition of new functionality to an existing service description. A typical case is the set of modifications required to the description of the basic circuit-switched bearer service (basic service) when a new supplementary service is standardized.

Where the description of a new service (e.g. a supplementary service) requires the modifications of the description of an existing service (e.g. the basic service), the SDL concept of a specialization can be used. A specialization enables the logic procedures associated with a new service to be described without affecting the description of the existing service by providing, for instance, the capability to:

– add and/or redefine block types in system diagrams;

– add channels in system type diagrams to convey existing and/or new signals;

– add and/or redefine process types in block diagrams;

– add routes in block type diagrams to convey existing and/or new signals;

– add new states and state transitions in process type diagrams;

– add new transitions triggered by new signals to existing states;

– redefine transitions of existing states; or

– redefine procedures.

In SDL, redefinition is enabled by defining as virtual those block types, process types, procedures and transitions that are likely to undergo change as the system evolves. Examples of virtual definitions are shown in Figures 10a, 11a, 12b, and 13.

2.5.2.2 Supplementary service description

A supplementary service is described by modifying or redefining the basic service logic. For this reason the basic service block types, process types and transitions that are likely to be affected by the addition of a supplementary service are marked "virtual" (see 2.5.2.1).

The changes to the basic service caused by the introduction of a supplementary service are then described by stating that the system "supplementary service" inherits the system type "basic service" and by adding new or redefined blocks, processes, transitions, signals, etc. as required to support the supplementary service. "Inherits" is a keyword in accordance with Recommendation Z.100

The description method is illustrated in Figures 14 to 17, using a simplified malicious call identification service as an example.

The system diagram of Figure 14 indicates that:

a) system type MCID is based on (inherits) system BasicService;

b) block BS_St2 of system BasicService has been redefined;

c) channel MCIDUserB has been added over which block BS_Stage2 exchanges signals MC_req, MC_conf, and MC_rej with the environment (user B in this case);

d) a new data type MC_rejcause has been introduced which can assume the indicated values (literals).

NOTE – Since it is necessary to show where channel MCIDUserB is attached, the instance of block BS_St2 has a broken outline to indicate that it is part of the inherited system, i.e. it is not a block that has been added to system BasicService for system MCID.

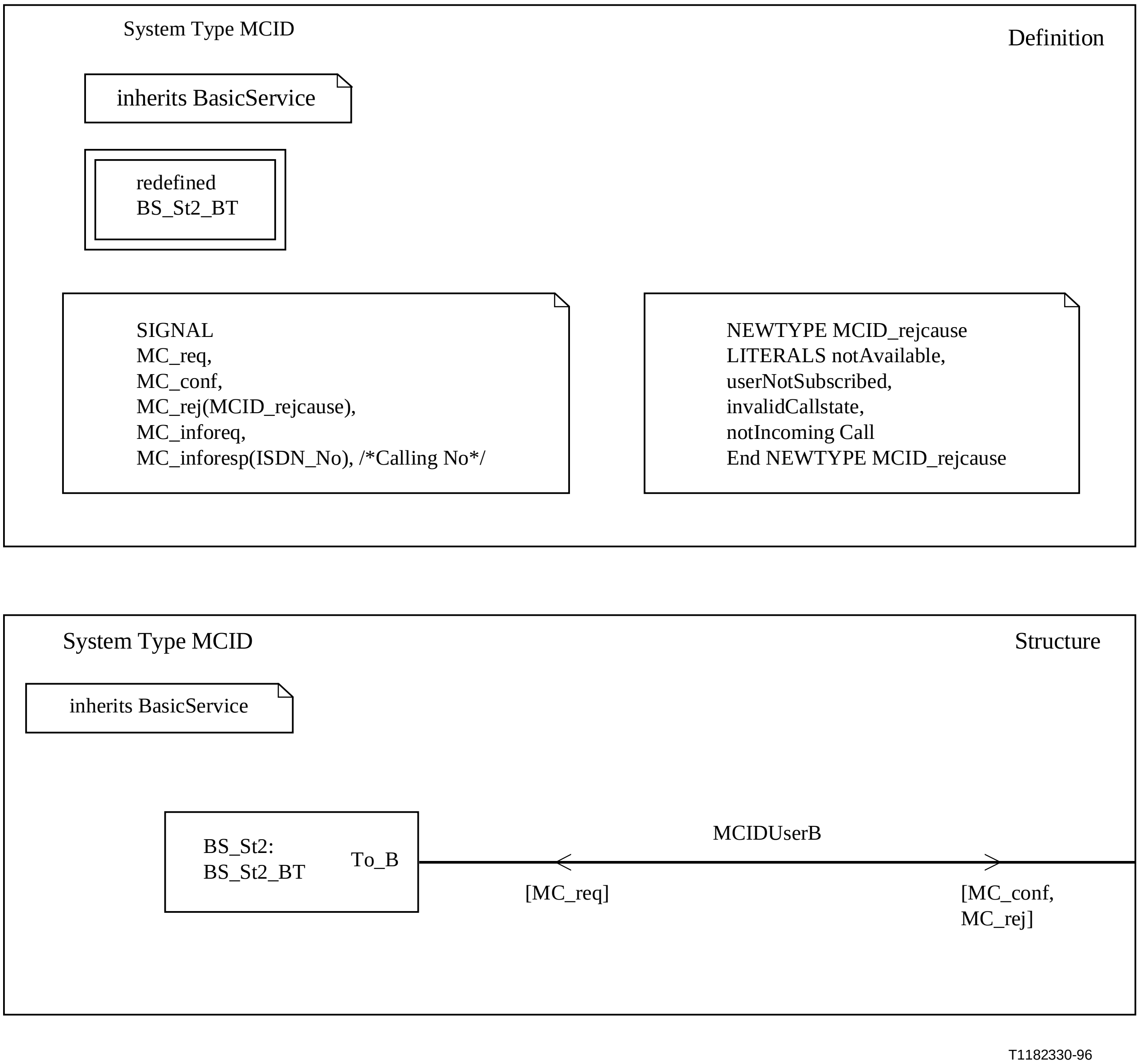


Figure 14/Q.65 – Example of a new system type (MCID) based on an existing system type (BasicService)

Figures 15a and 15b show the redefinition block type BS_Stage2_BT caused by the introduction of the MCID supplementary service. It required the addition of a process type MCID_PT and of routes RA, RC1, RC2, and MC_UIB as well as the additional signals contained in the signal lists of Figure 15.

Process types shown with broken outlines are not part of the redefinition but are shown so that it can be indicated where the added routes are attached.

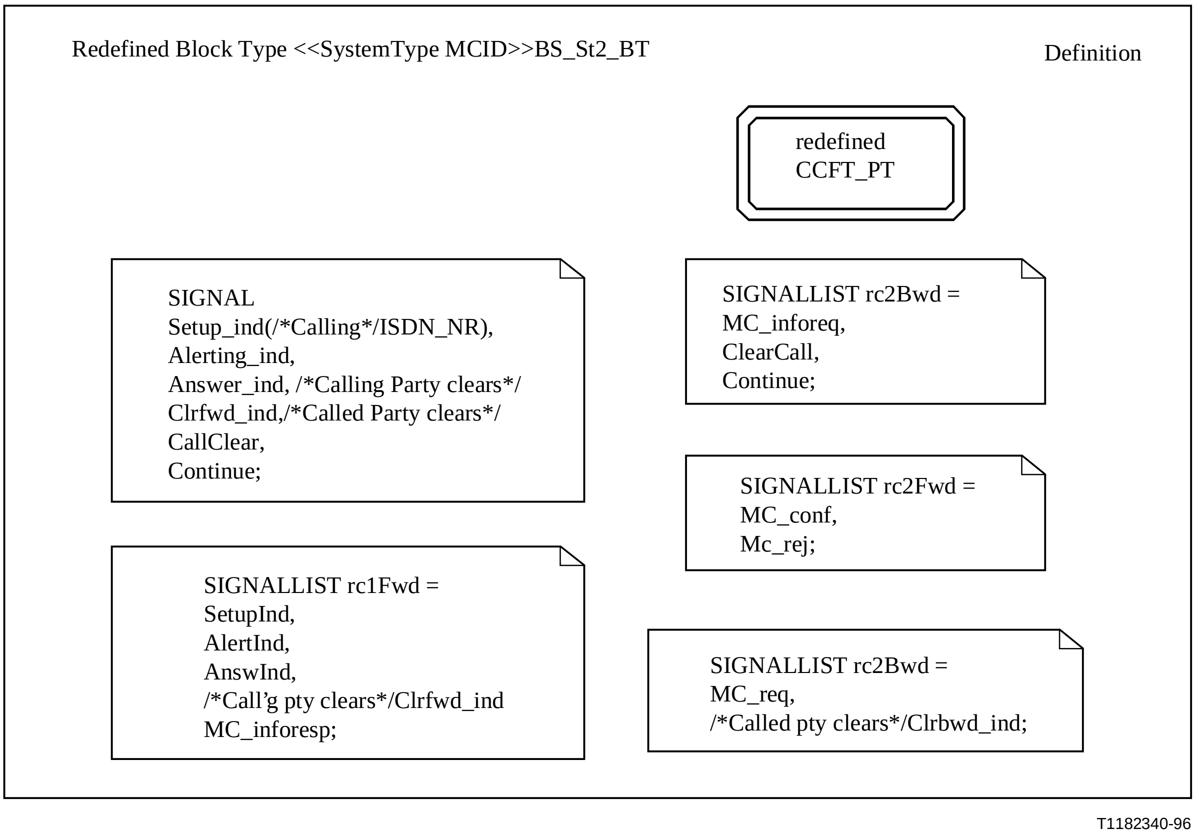


Figure 15a/Q.65 – Block Type BS_St2 (Definition) redefined for System MCID

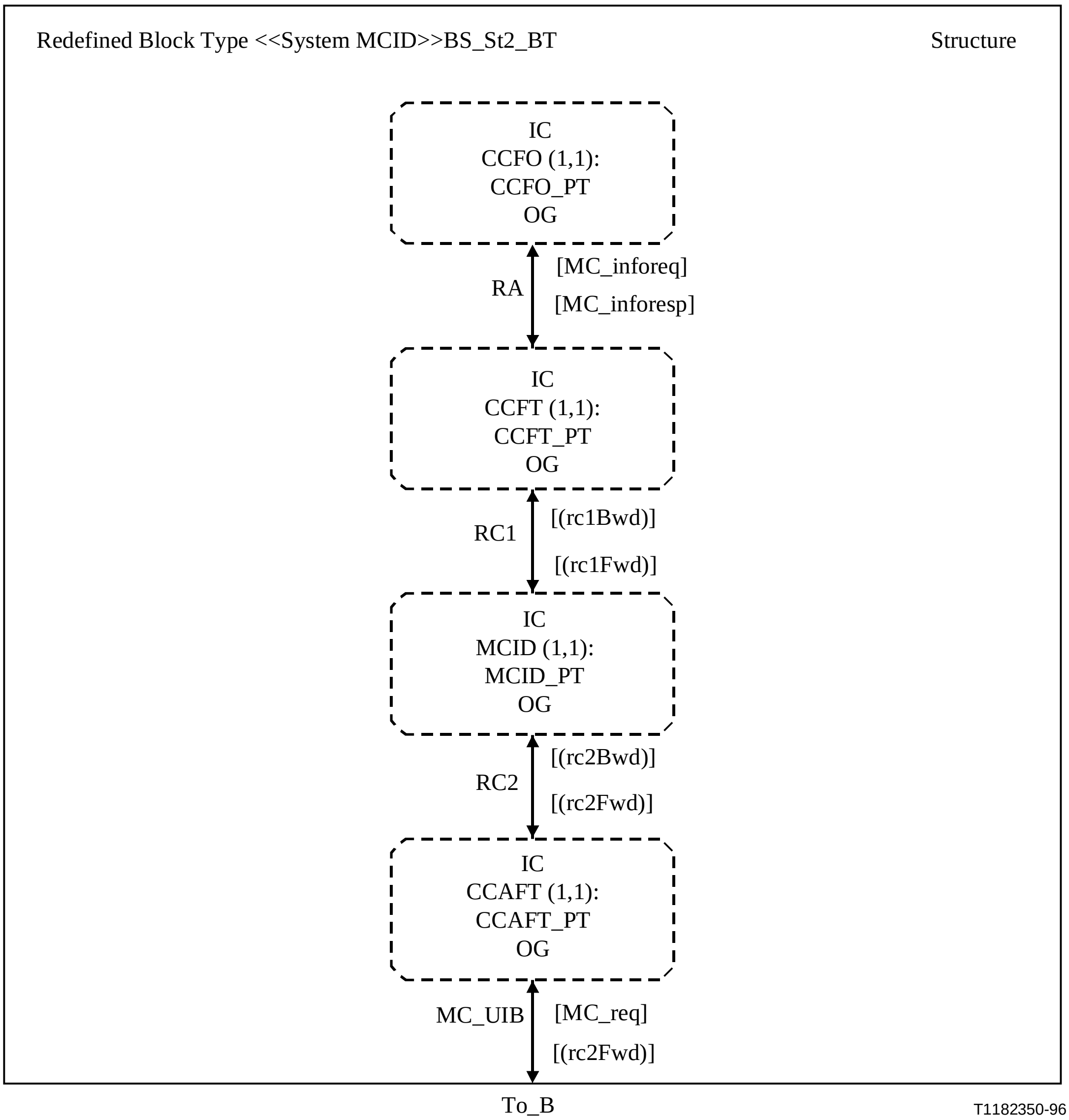
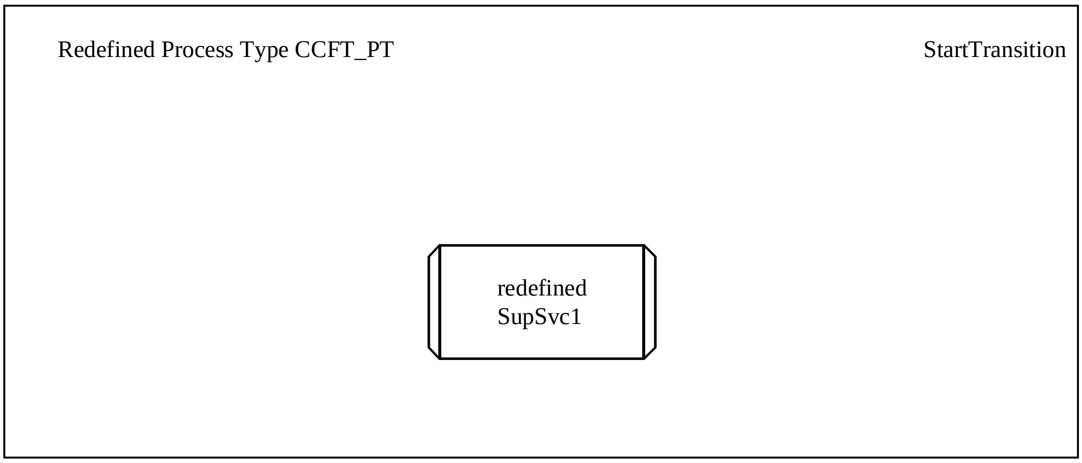


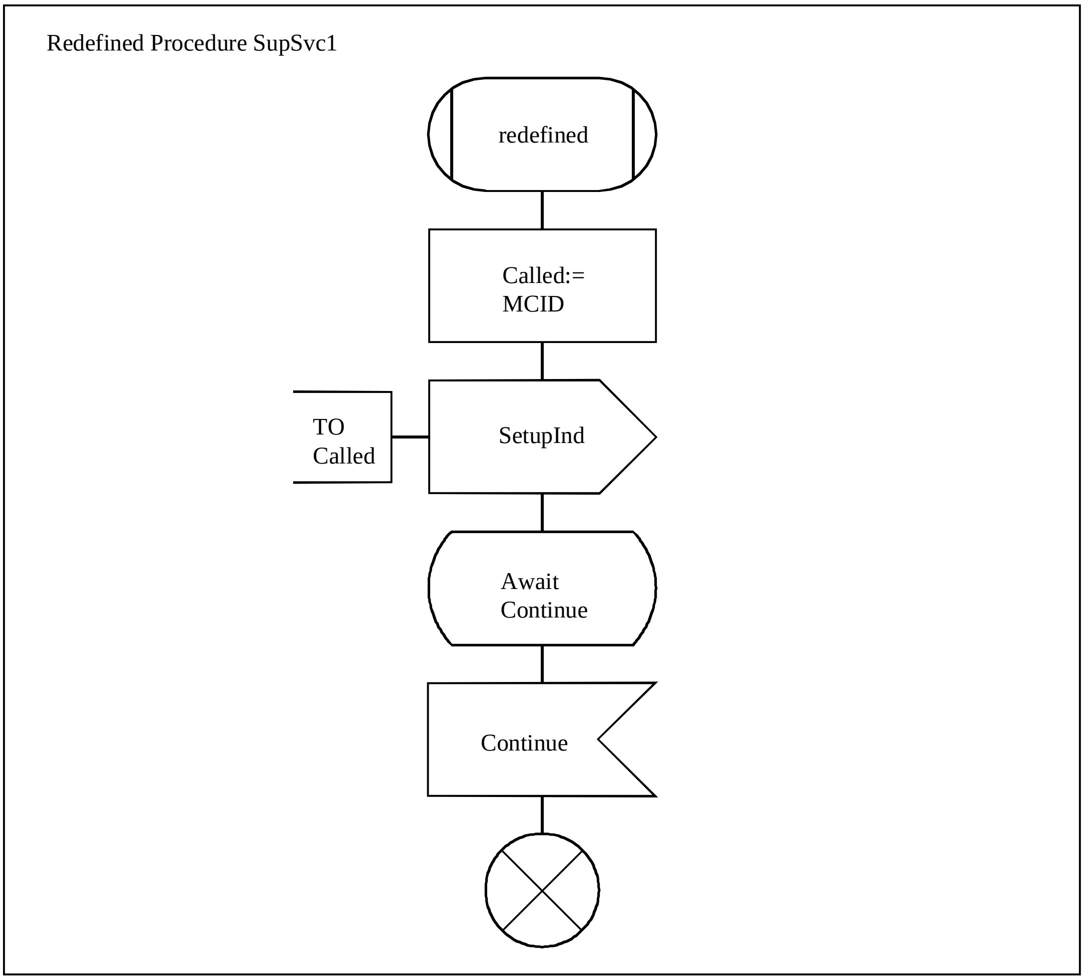
Figure 15b/Q.65 – Block Type BS_St2 (Structure) redefined for System MCID

Figure 16 shows the redefinition of process type CCFT_PT. The only change that needs to be shown here is the indication that procedure SupSvc1 which is defined as virtual in system type BasicService is being redefined. The redefined procedure is shown in Figure 17.



T1182360-96

Figure 16/Q.65 – Process Type CCFT_PT redefined for System MCID



T1182370-96

Figure 17/Q.65 – Procedure SupSvc1 redefined for System MCID

2.5.3 Packages

SDL 92 offers the possibility to use types defined in one system in other systems as well. This is achieved by providing a package diagram which serves as a container for items such as system type definitions, block type definition, process type definitions, signals or synonyms.

The use of a type which has been defined in a package is indicated by providing, in a text symbol above a diagram, the name of the package preceded by the keyword use. An example of a package definition and the use of a type defined in the package is shown in Figure 18. The figure reflects the fact that block type B1 in system BCSystem is defined using block type Block1_BT in package BCBlocks.

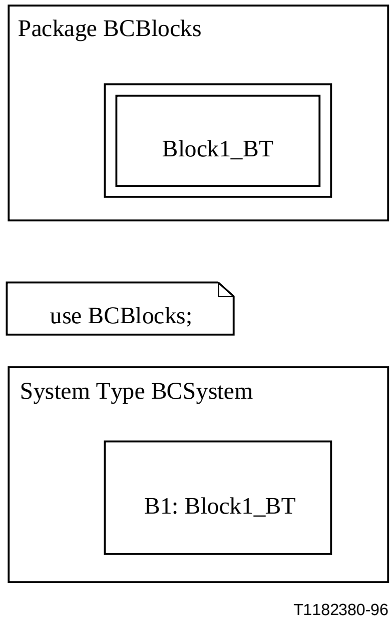


Figure 18/Q.65 – Example of the definition of a block type in a package and the subsequent use of the package in a system

2.6 Step 6 – Allocation of functional entities to physical locations (scenarios)

In Step 1, a functional model consisting of functional entities, each of which has a well-defined relationship to the others is defined for each basic and supplementary service. Step 6 is the allocation of these functional entities to physical locations and defines all relevant physical implementations, henceforth called scenarios.

More than one scenario may be defined for one functional model so that Administrations will have options as to where the service is actually provided. For example, a supplementary service functional entity could be located whether in a PBX or in an exchange.

For the allocation of functional entities, it should be noted that:

a) a functional entity may, in principle, be allocated to any physical location;

b) a number of functional entities may be allocated to the same physical location;

c) for every supplementary service, network scenarios which include the location of its basic service functional entities should be defined;

d) different physical locations of functional entities may imply minor differences in node capabilities (e.g. the transmission path switch-through actions may depend on whether the access is in an exchange or a PBX);

e) the relationship between pairs of functional entities, according to the functional model used, should be invariant for all of the recommended scenarios.

Item e) implies, for example, that the information flows from a supplementary service would not be affected by a reallocation of one or more of the required functional entities from public network exchange to a PBX or vice versa.

All identified scenarios will be considered in Stage 3 for definition of signalling protocols, switching capabilities and service capabilities.

Possible physical locations and their corresponding symbolic representation are:

– Terminal equipment: Type 1 or terminal adapter TE;
– Network termination; Type 2: NTE (typically in PBX);
– Local exchange: LE;
– Transit exchange: TR;
– Service switching point: SSP;
– Service control point: SCP;
– Database: DB;
– Intelligent peripheral: IP.

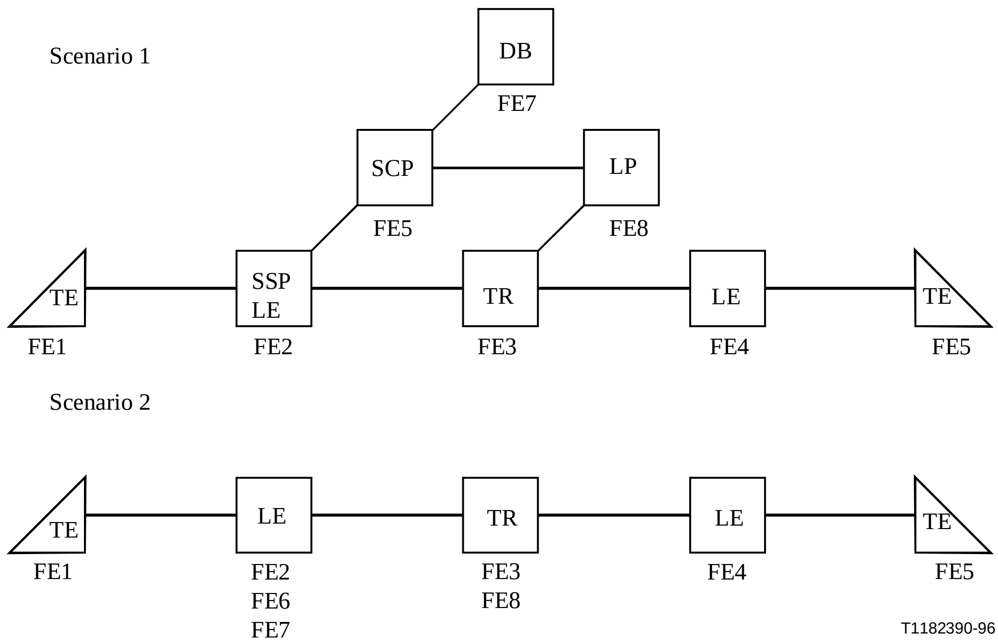


Figure 19/Q.65 – Example of two scenarios for assigning functional entities to physical locations

APPENDIX I

Format and outline of a Stage 2 description using the unified functional methodology

1 Scope

2 Normative references

3 Definitions

4 Symbols and abbreviations

5 Description

6 Derivation of the functional model

6.1 Functional model description and relationship with the basic service

6.2 Description of functional entities

7 SIB-based service feature definitions

8 Information flows

8.1 Information flow diagrams

8.2 Definition of individual information flows

8.2.1 Relationship r1

8.2.1.1 Contents of information flow

8.2.1.x Contents of information flow

9 Functional entity actions

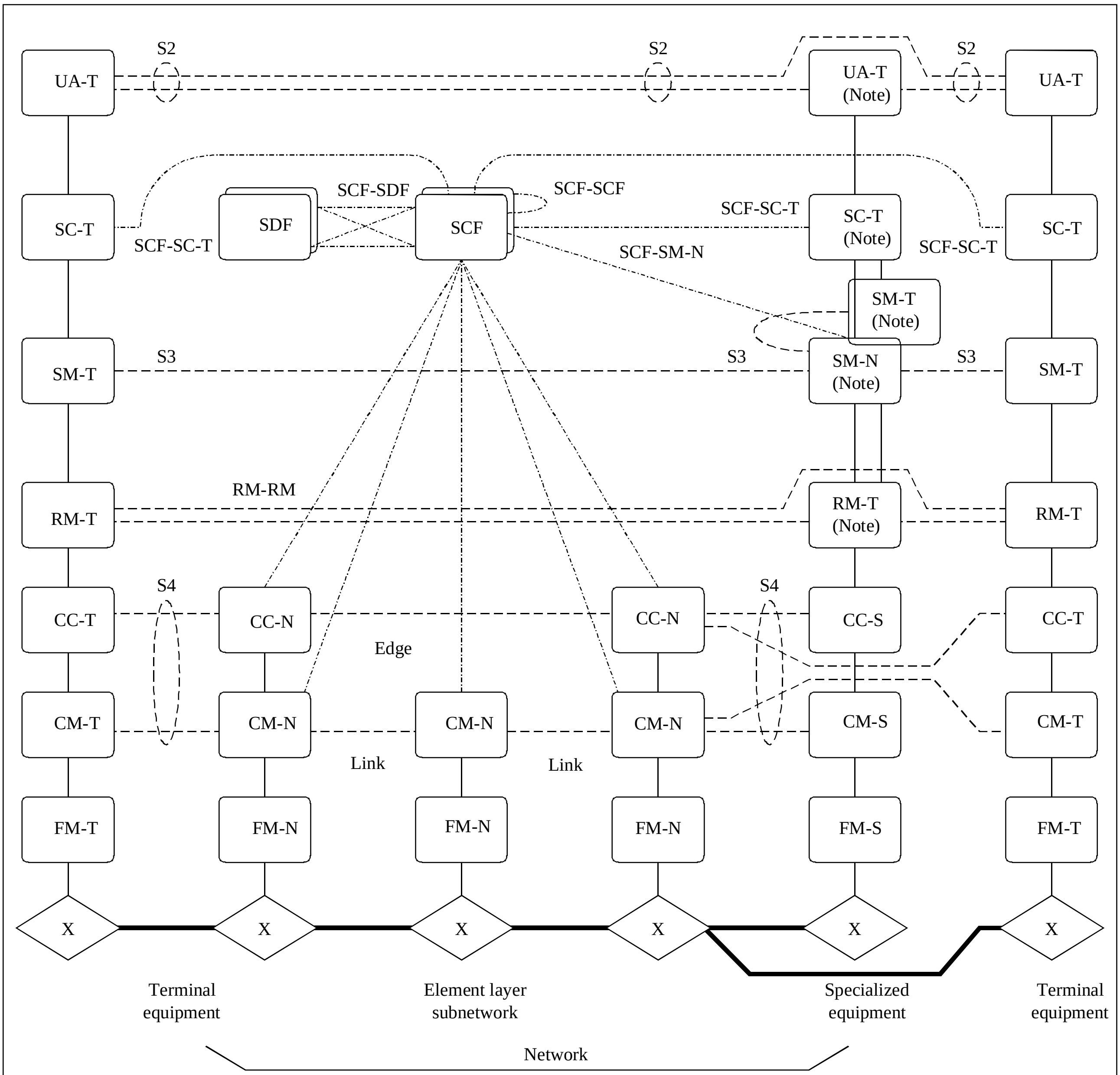
10 SDL diagrams for functional entities

11 Allocation of functional entities to physical locations (scenarios)

APPENDIX II

Functional architecture – Q.65 evolution (for future study)

This diagram presents a functional entity model without reference to any physical realization or architecture. The "CM-N" entity shown corresponds to just the real-time connection control component of the NML shown in the unified functional architecture. The "FM-N" entity shown corresponds to just the real-time resource and fabric control of the EML shown in the unified functional architecture.



Physical connectivity

Peer-to-peer signalling relationships

IN signalling relationships

Inter-service level signalling relationships

UA = user application function, SC = service control function, SCF = service control function, SDF = service data function
SM = session management function, RM = resource management function, CC = call control function,
CM = connection management function, FM = fabric management function

NOTE – These functions could be emulated by the SCF; in such cases, the relationships towards these functions would lead to the SCF.

S2 signalling is carried by user SVC, S3 signalling may be carried by signalling SVC or user SVC, S4 signalling is carried by signalling SVC.
Specialized equipment may be attached to SN (serving node) or gateway nodes.

T1183760-97

Figure II.1/Q.65 – Unified functional model

ITU-T RECOMMENDATIONS SERIES

Series A	Organization of the work of the ITU-T
Series B	Means of expression: definitions, symbols, classification
Series C	General telecommunication statistics
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	TMN and network maintenance: international transmission systems, telephone circuits, telegraphy, facsimile and leased circuits
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	**Switching and signalling**
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks and open system communication
Series Z	Programming languages