ptt research

Graduation Report

Broadband ISDN signalling release 2/3
Stage 2 development

R.W. Gerrits

Eindhoven University of Technology
Faculty of Electrical Engineering
Digital Information Systems Group

Leidschendam, June 3, 1993

Graduation professor:	Prof. Ir. J. de Stigter
Supervisors:	Ir. P.F.C. Blankers
	Ir. H.J.H. Beckers
Graduation period:	July 1, 1992 - June 10, 1993

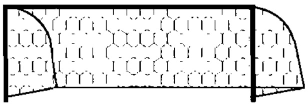
The Faculty of Electrical Engineering of the Eindhoven University of Technology does not accept any responsibility regarding the contents of student graduation reports.

Broadband ISDN signalling release 2/3
Stage 2 development

Volume I

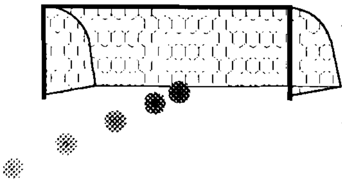
R.W. Gerrits

Preface



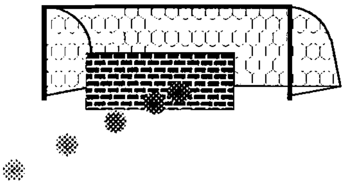
Somewhere in the Dutch history a guy started to study Electrical Engineering. I don't mention his name. I know, you don't care.
The fellow – let's call him R – had some kind of goal in his mind, something like: "Whow, I want to learn EVERYTHING about those dinky electrons. This must be great!"

By the way, if you say to yourself "Gush, do I feel sleepy.... (some long lasting noise)", then I can better warn you in advance: This bunch of paper may be less somnolent if you turn over this page.



After a couple of lectures and a number of years (time flows) R has made acquaintance to many other 'electrical people' – let's call them students at the Eindhoven University of Technology. He even learned quite a lot jargon: dinky electrons became nearly infinitesimal quantum mechanic objects. Help! Electrons do not exist! They're just a hotchpotch of quarcks.
Fortunately R discovered that electrons still appear to flow around in his television. So he ploughed through every lecture to reach his goal: get that diploma.

But many obstacles arise on the way to become an electrical engineer. The graphical translation speaks volumes, I suppose.
No.. not the one up here but this picture ⟶



The end comes close, fortuitously. R started his graduation task. A great time out there in Leidschendam (Mommy, where is the map) happened to be the consequence. How time flies!
Now I'm writing this piece of text, thereby finishing my graduation task.
Yeah, I know, you found out the mistery R: "He tried to keep it a secret to the end, but I've beaten him."

I'm waisting paper. Let's get serious!
A lot of people have supported me during my graduation task. I have made acquaintance to many colleagues during the last twelve months. I'm anxiously awaiting phone calls of people I

forgot to mention. Feel free to append your name to the list! After you have perceived my warranty disclaimer, I invite you to read the rest of the page.

First I want to thank my family for patiently waiting those many years till the moment arises that I graduate. Especially my girl friend, Angela, for supporting me several years. She gave me plenty hearings about stage-2 stuff lately.

I want to thank my friends in Eindhoven and in the Dutch army for helping me to get things arranged during my residence near Leidschendam. Before I go deeper into Leidschendam, I must mention Rian van Gaalen, secretary of the Digital Information Systems Group, for taking care of the 'final touch.'

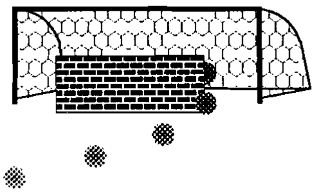
We're almost in Leidschendam... we went by car. Okay, I didn't drive. Wim van Berchum has brought me home every weekend, and vice versa each Monday (even before sun rise).

Now we enter the building of PTT Research...
First I want to thank Patrick Blankers and Harry Beckers for supervising my thoughts and work from the first day up to now. Also my graduation professor J.de Stigter taking up the challenge to get me graduated.
I'm told that it is *very wise* to thank your boss. Therefore I thank Jan van Loon for not letting me minute any meeting (and for not getting angry after reading this, hopefully).

I want to thank all employees of Signalling – oops, Network Control – project group for dealing with my questions on B-ISDN signalling.
In this place I want to nominate Pascal Heynen for the personal debugger award (never heard of?), because he greatly helped me to get the program Flows what it has become. In this light I have to thank Guy Reyniers for the many tips.

Finally I thank Murphy for not striking the VAX on the right moment I am printing this document.

Rob Gerrits
Leidschendam, June 3, 1993



So in a roundabout way I am reaching my goal.

Abstract

In the development of B-ISDN several aspects can be distinguished, like functional description of the signalling system, definition and description of services, data transmission protocols, as specified by the CCITT three stages characterization methodology. Furthermore B-ISDN signalling will be introduced in three releases.
This report deals with the development of functional network aspects of B-ISDN release 2/3.

First the signalling system of B-ISDN based on separation between call control and connection control is explored. The benefits and drawbacks of the two level system are identified. Also the state of the art at the beginning of this graduation task are signed up.

The following step describes several functional models which are the basis of functional network description. After weighing the models against another, the three level model with an intermediate Special Resource Control level appears to be too complex as a rudiment for the functional description of the B-ISDN network. The proposed AC model seems to have a strong hand.

The major part of this report explores scenarios by means of information flow diagrams. Two party calls and multiparty calls pass in review. Both point-to-point and point-to-multipoint with branching nodes are described utilizing the separated two level model. Multiparty calls having a non-distributed bridge are described in a complex three level. Also the inner of functional entities and information flows are studied.

Ensuing the stage 2 characterization methodology, a SDL diagram should be derived. The SDL diagram developed as part of the task shows a significant difference in complexity between the CC and BC level. It also shows that branching supported by the BC level add just a small amount of complexity.

Finally the functional specification of a Q.73 demonstrator has been elaborated. This program will simulate information flow diagrams developed for B-ISDN release 2. The implementation is partially fulfilled within this graduation task.

Contents

Volume I

Volume II

Appendices

1 Introduction

1.1 Integrated Services Digital Network

The public telephone network is excessively digitalized nowadays and the introduction of the Integrated Services Digital Network (ISDN) is at an advanced stage. An ISDN provides a set of digital speech and non-speech services based on a limited set of user-network interfaces (UNI's) and network-node interfaces (NNI's). The ISDN supports both circuit and packet-switched connections.

A distinction is made between narrowband and broadband ISDN. Narrowband ISDN (N-ISDN) is the network that has been introduced already. It has a restricted capacity and functionality. On the other hand Broadband ISDN (B-ISDN) will be the network with a major capacity and functionality. This report will deal with an aspect of the B-ISDN signalling development. Signalling encloses all information exchange to control the network resources required to offer any service.

In the development of B-ISDN several aspects can be distinguished, like functional description of the signalling system, definition and description of services, data transmission protocols, et cetera. Furthermore B-ISDN signalling will be introduced in three releases.

The first release (standards to be completed in 1993) uses a monolithic signalling protocol, which means that a one-to-one relationship exists between “call” and “connection.” Release two will be characterized by a separation of the notions “call” and “connection,” thereby introducing the possibility of having a call with more than one connection involved. Chapter 2 will digress on the theory about this subject. Also release two will support more supplementary services. The standards for this release is planned to be ready in 1995. The third release, called target B-ISDN, will support extended multi-media services and broadcast connections (ready in 1997). This report will go into the functional description of the signalling system of B-ISDN release 2 and 3. A functional description delimits entities in a system and describes the way these entities interact.

International standardization bodies, like ETSI and CCITT, pay much attention to the development of a functional description of the control system for the future broadband ISDN. The development of this description is still in an early stage. Up to now the functional description encloses a functional model and a first set of information flow diagrams published in baseline documents. No released recommendations or standards exists at present. The functional description is based on separation between call control and connection control (see chapter 2), and will be the foundation for the future development of signalling protocols.

1.2 Targets of the Graduation Task

The graduation task includes three targets. The first target is the development of a functional description of the basic call for B-ISDN release 2/3. The second target is to design and implement a demonstrator for the signalling system. The requirements of this demonstrator are described below. The final target includes the investigation of the correctness of the results (ETSI and CCITT results and the results of this graduation task) and, if necessary, these findings will be presented in ETSI and CCITT.

The demonstrator must meet the following requirements:

- The demonstrator must show how a control system with separated call control and connection control operates;
- The demonstrator must show the benefits of the separation between call control and connection control;
- The demonstrator must be easy to work with;
- Several scenarios must be demonstrated, e.g. call setup, connection setup within an existing call, simultaneous setup of a call with one or more connections;
- The demonstrator must be easily extendable for new scenarios.

1.3 Outline of the Report

The activities to attain the targets of the graduation task may be grouped in three sets. The first set encloses the (partial) development of the stage 2 functional description. At first new information flow diagrams must be developed, e.g. for multi-connection calls and multi-party calls. Next SDL[1] descriptions of the functional entities in the model must be derived from these information flow diagrams. Chapter 4 and 5 will go into these subjects. Only switched on-demand services in connection-oriented mode will be discussed in this report.
The second set encloses the design and implementation of the demonstrator. This program will be discussed in chapter 6 and – mainly – in [17].
The third set includes the testing of scenarios developed by ETSI and CCITT and by the author. Most likely this will result in a number of contributions to ETSI/CCITT.

The following notational conventions will be pursued:
Information flows appear in sans-serif condensed typeface.
Pascal procedure headers appear in `monospaced` typeface.
SDL signals appear in **Times bold** typeface

[1] SDL is a standard formal language for the specification and description of systems. It has been standardized by the CCITT. See also chapter 5.

2 Broadband ISDN Signalling

2.1 Why New Signalling Systems

The demand for multiparty-, multimedia-, and mobile telecommunication services will increase fast in the near future. The introduction and integration of these new services will have major consequences on the telecommunication networks. The B-ISDN network based on ATM offers a lot of flexibility: ATM supports a wide range of services. However, this flexibility can only optimally be used and these services can only be offered if the signalling system is flexible as well. The current signalling systems (DSS1, C7) are not able to support these services without due consideration. The control of the services must be revised. This fact requires a new understanding of the notions "call" and "connection" and the introduction of the principle of "the separation between call control and connection control" in the telecommunication networks. This chapter will go into several aspects of the development and introduction of B-ISDN. As will be explained a stepwise method will be used as a structured way to realize B-ISDN. Only switched on-demand services in connection-oriented mode will be discussed.

2.2 Call and Connection

It is important to define call and connection to prevent any indistinctness.

Definition of Call

"An association between two or more users or between users and network entities, that is established by use of network capabilities to provide a telecommunication service. This association may have zero or multiple information exchange mechanisms established within this call, for example in connection-oriented or in connectionless modes."

Definition of Connection (1)

"An end-to-end association between two or more parties, that provides the transport of user data."

Definition of Connection (2)

"An end-to-end association between two or more parties, that provides the transport of user data without processing the user data."

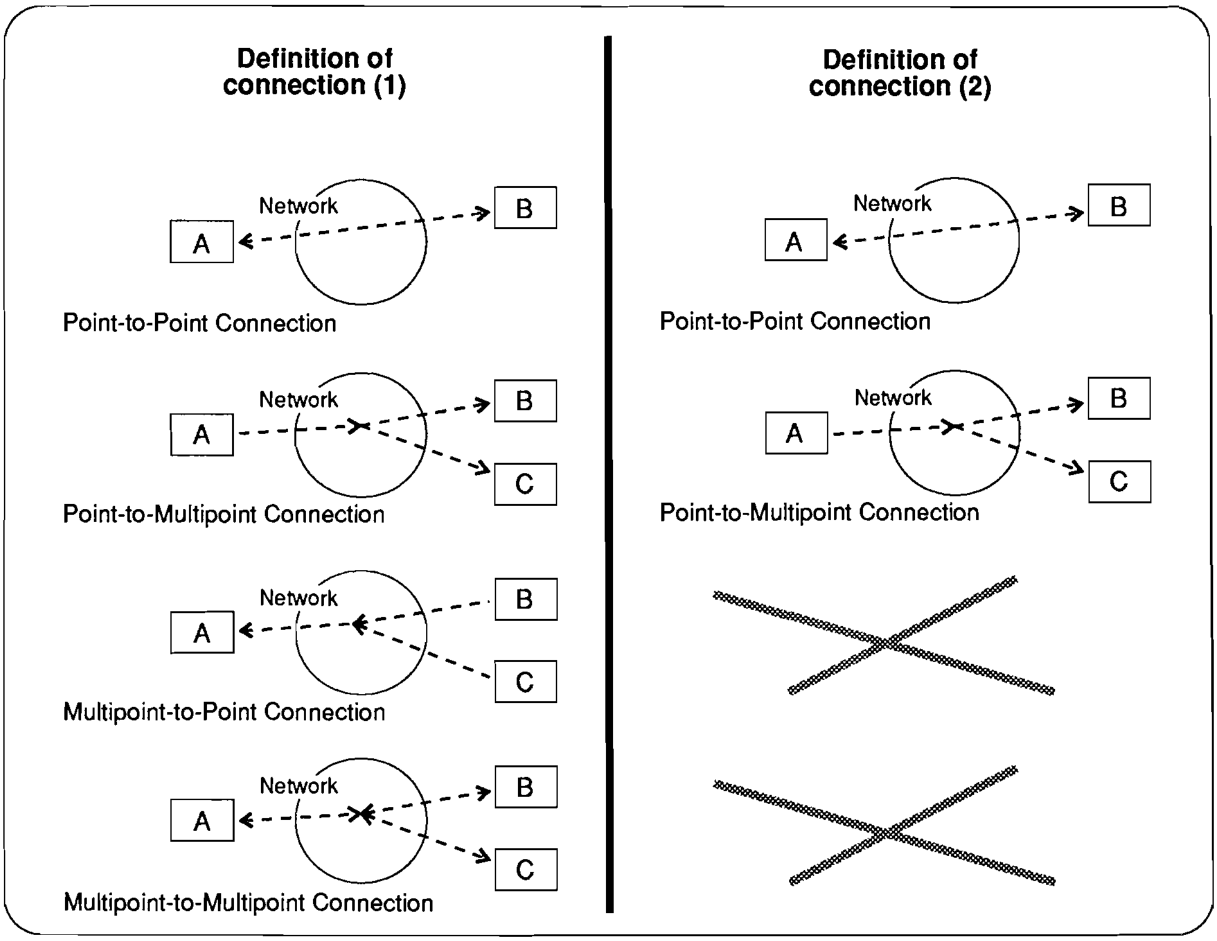

Definition of connection (1)
Definition of connection (2)
Network
A
B
C
Point-to-Point Connection
Point-to-Multipoint Connection
Multipoint-to-Point Connection
Multipoint-to-Multipoint Connection

Fig. 2.1 Differences between the two definitions of Connection.

Connections may be uni-directional as well as bi-directional. The differences between the two definitions of Connection are illustrated in figure 2.1. The first definition is used by the CCITT [9]. The second definition will be used in this report, because it is less confusing:
Definition 1 allows the insertion of a bridge in the connection, as will be used in a "conference call" to mix and distribute the user data. Definition 2 does treat a bridge as the endpoint of a connection. So all incoming user data (from the bridge point of view) are different connections as well as the outgoing user data. Figure 2.2 show this distinction. In this figure, the bridge is a functionality that mixes the incoming user data (e.g. video or audio) and distributes the result to all involved users.
In the lower half of figure 2.2, the bridge is an endpoint for all connections, unlike the upper half according to definition 1.

2.3 CC and BC

The formal definition of Call Control (CC) is: "CC controls the call association configuration between two or more paries and coordinates zero, one, or more bearer control entities between these parties."
The formal definition of (Bearer) Connection Control (BC) is: "BC controls the bearer control association configuration between two or more parties and establishes, modifies, and disconnects the bearer/ATM connection between these parties."
The functions CC and BC perform are specifically displayed in table 2.1.

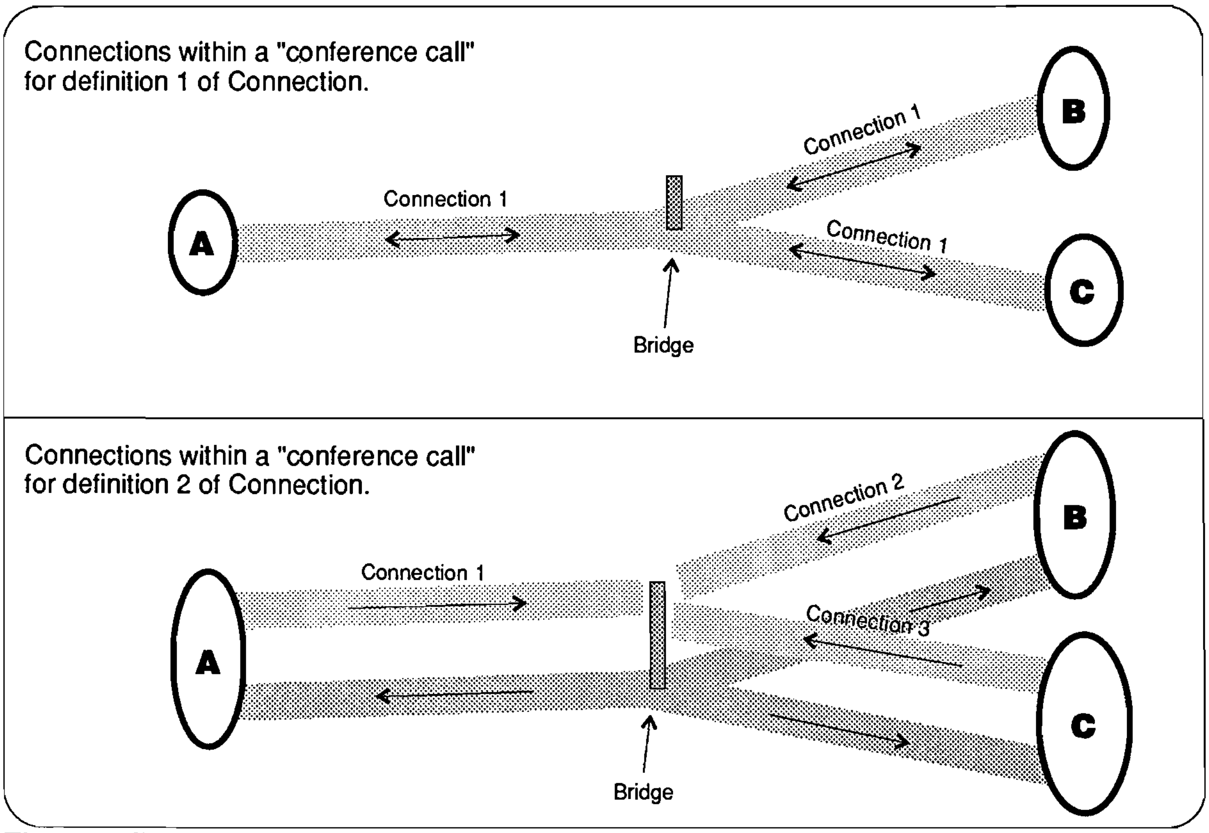

Connections within a "conference call" for definition 1 of Connection.
Connection 1
Connection 1
Connection 1
A
B
C
Bridge
Connections within a "conference call" for definition 2 of Connection.
Connection 1
Connection 2
Connection 3
A
B
C
Bridge

Fig. 2.2 Distinction between both definitions of Connection in case of a "multipoint-to-multipoint" connection.

The concept of separation between CC and BC in the control system of a telecommunication network has the consequence that the functionalities of CC and BC are specified separately in the network. However they may be positioned in the same physical location. The current signalling systems have no separation between CC and BC. These systems are called monolithic, because the functions of CC and BC are strongly interweaved. For future signalling systems the separation between CC and BC will be one of the most important features.

So the BC level deals with connections only. If one and the same user data stream has to be delivered at more than one destination, the stream must be split into two or more data streams somewhere. This is called multicasting.
The CC level deals with associations, user requests, and telecommunication services. The CC level will interchange information with the BC level, the human user, and possibly other functional entities. A functional entity (FE) performs a delimited set of functions and actions by controlling resources and interchanging information with other FEs. If two FEs communicate to each other, a relationship exist between these entities. The formal definitions of FEs and FE relationships can be found in appendix A.

2.4 Separation Between CC and BC

It shall be clear that one call – one association in the CC level – may involve more than one connection in the BC level. A potential configuration may be a call between two parties involving a 3.1 kHz audio connection and a 64 kbit/s digital connection, in case of a videophone conversation. The major difference between a separated signalling system and the monolithic system, as modelled in recommendation Q.71 [2], can be found in this feature.

Table 2.1 Functions of the CC and BC levels.

Call Control (CC)	Connection Control (BC)
Overall control of the call	**Overall control of the connection**
- setup/release of the association(s) between CC entities	- setup/release of the association(s) between BC entities
- selection of next CC node	- routing (selection of the next BC node/switching)
- identification of association(s) between CC entities	- maintenance of the state of the connection
- maintenance of the state of the call	- identification of the connection at the BC level
Service admission control	**Control network resources**
- telecommunication service screening	- reservation/release network resources
- user authentication and authorization	- allocation/deallocation of network resources
- service component negotiation	- through-connect/disconnect
- user status checking and monitoring	- (re)negotiation of network resources
- (re)negotiation of end to end QOS	- interworking with existing networks
- compatibility checking	- modification of control 'rights' to BC entities
- modification of control 'rights' to CC entities	
- interworking with existing networks	**Communication with CC**
	- notification of bearer progress
Communication with BC	- delivering charging information
- give order to setup and release connections	- notification of change in bearer status (connected, failure, ...)
- coordination between connections	
- identification of connections at CC level	
- collection of charging information	

Another profit is the possibility of coordination between connections by the CC level. The monolithic signalling system has only one level, which supports only one connection. Speaking in terms of Q.73, a call must be set up for each connection. So no relationship exists between two connections. This fact excludes almost any possibility of shared conditions like common identifiers (e.g. in case of videophony, forcing connections to be set up via the same path through the network and synchronize them).

Except the above mentioned benefit, some other enhancements characterize a separated control system. Considering a three party video conference call, another enhancement will get plain. In the CC level one relationship exist, so we're talking about one single call. This relationship involves the three parties and a bridge functionality somewhere in the network. The BC level will be organized in a different way: the connections will not exist between all parties, but between the bridge and each party. The bridge mixes the incoming speech and video and distributes the result to all parties. Figure 2.3 shows clearly the difference between the CC level and the BC level.

Having a monolithic signalling system, first a call must be set up between the first party and the bridge, whereupon the bridge will set up calls to the other parties. So no relation exists between two parties, which results in the impossibility to have direct user-to-user signalling.

It is possible to interchange signalling information without setting up a connection. This enhancement of signalling system based upon a separation between CC and BC opens the possibility of setting up a call and a connection sequentially. First the call will be set up. If the other party has a compatible terminal which is idle, the connection will be set up. So no unnecessary bandwidth will be reserved or even allocated if the other party has no free compatible terminal.

Because the CC FEs may be absent in transit exchanges, less processing will occur for end-to-end information. So this information will travel faster through the network, which is another enhancement of the separated system. Of course the transit exchanges can be more simple and need less capabilities.

Because a different state machine appears for every connection, clear distinction exists between all connections. So a clear signalling can be achieved without getting confused about the condition of a connection.

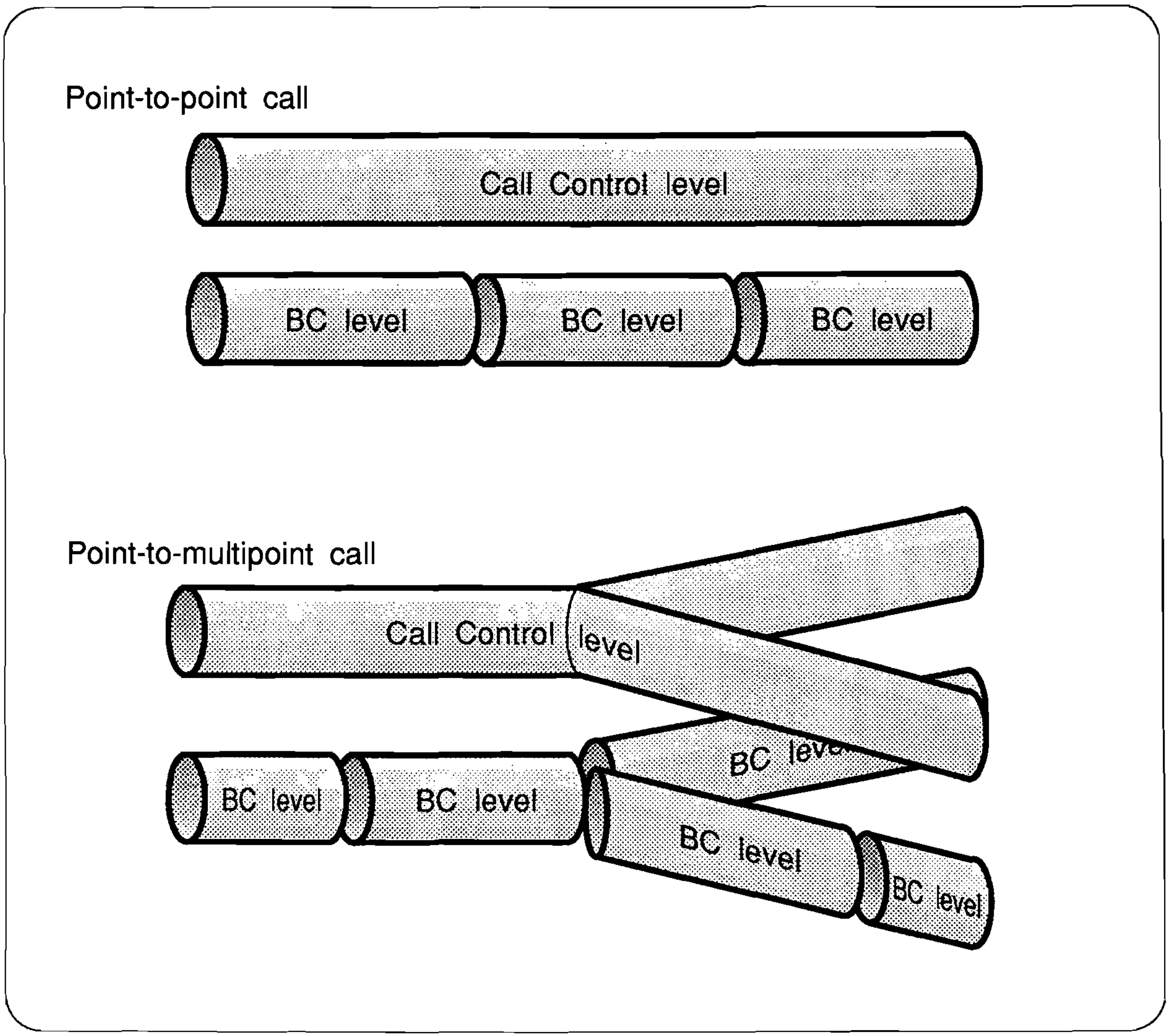


Fig. 2.3 Difference between CC level and BC level associations: CC level uses end-to-end and BC uses link-by-link association.

Conclusions: The separation between call control and connection control results in a large improvement of the flexibility of the signalling system.
As every light has it's shadow, a price must be paid for a separated control system. The system will be characterized by a more complex functional description, more complex protocols, and lots of investments and manpower to standardize and develop this system.

2.5 B-ISDN Introduction in Three Releases

At present ISDN, or more precisely N-ISDN, is about to get operational on large scale. The N-ISDN gives a subscriber the possibility to make use of a large amount of telecommunication services via one single connection. The N-ISDN is an enormous improvement with regard to the current set of service specific networks. Anyhow the end of the developments in the public telecommunication network will be still out of the question. The next step is the development and introduction of B-ISDN.
The B-ISDN is not a simple extension to the N-ISDN. In fact it will differ substantially in almost any respect; not only the bandwidth and speed, but also the data transfer mode, services, the control system, et cetera. These changes will have a great impact on the public networks. To meet the changed needs, two possible ways can be followed: adjust the current

systems or redevelop and replace the current systems. Adjustment might work in short term, but will not be the solution for the long term. So the CCITT decided to standardize a long term, broadband public telecommunication infrastructure, called the target "broadband ISDN," which is based on data transport via ATM [5]. It must meet all the future needs.

It would be an illusion, that it might be possible to introduce a entire new B-ISDN control system at once. That's why much attention is paid internationally to the so called "evolution scenario." It describes how the current network and control systems should migrate to the target B-ISDN signalling system. For B-ISDN control systems first a minimum set of services will be identified that will be extended in each following phase. CCITT study group XVIII has developed a table which distinguishes three releases [4]:

- release 1: A short term solution;
- release 2: A medium term solution;
- release 3: The 'target' solution.

The standards for release 1 signalling are planned to be ready in 1993. Releases 2 and 3 will follow two respectively four years later on. It's important to realize that release 1 and 2 are on standards on themselves, because the final aim is target B-ISDN. Compatibility between the subsequent releases is of major importance for the evolution scenario.
Release 1 will still be similar to N-ISDN, but the ATM [5] concept for user data transport will be the most important feature of this release. New in release 2 are multi-point connections, multiconnection calls and look-ahead procedures. New in release 3 are broadcast connections. Of course each release supports a superset of services according to the prior releases.

2.5.1 New Requirements to Signalling

Several factors are the course of a number of new requirements to the B-ISDN signalling systems. The origin of the most important requirements may be divided in four categories:

- Requirements originating from the telecommunication services, like: distributive services (radio, television), point-to-multipoint configurations (video conferencing), multimedia services (combinations of speech, video and data), bursty traffic (file transfer);
- Requirements originating from the transport network for user data: the choice for a total new transport network introduce new requirements to B-ISDN signalling, because ATM channels must be controlled in stead of fixed 64 kb/s circuits;
- Requirements originating from the transport network for signalling;
- Other conditions, like compatibility requirements for subsequent releases of signalling protocols.

The need of flexibility *in the signalling systems* will be met by separating CC and BC and by introducing a more modular structure.

2.6 Three Stages Methodology for Characterization

Developing a new signalling system isn't that simple, so ad hoc development has a great chance to terminate early. The characterization of telecommunication services supported by an ISDN and network capabilities of an ISDN can be divided into three main stages of activity [3,10]:

- stage 1: Service aspects;
- stage 2: Functional network aspects;
- stage 3: Protocol specification aspects.

Within each stage a number of steps have been identified. In principle, the application of the method is sequential, stage 1 giving the service description from the user point of view, stage 2 offering an intermediate view of what happens at the user-network interface and inside the network between different exchanges, and stage 3 giving the actual switching and service nodes descriptions, as well as protocols and formats to be adopted.

The application of the method for stage 1 results in a description of the service. Stage 2 results in one or more implementation independent scenarios, and stage 3 results in a set of protocol and switching recommendations needed to realize the service for each scenario.

Stage 2 has been subdivided into five process steps as can be seen in figure 2.4. These steps may be passed through several times, like an iterative process, to complete a stage 2 functional description of a signalling system.

2.7 State of the Art

2.7.1 Stage 1 B-ISDN

The stage 1 descriptions of services from user point of view enclose for each release a number of services. The descriptions are taken from a set of draft recommendations coming from CCITT study group I, like:

- Broadband connection oriented bearer service (draft rec. F.811);
- Broadband connectionless data bearer service (draft rec. F.812);
- Broadband videotelephony (draft rec. F.722);
- Broadband videoconference (draft rec. F.732);
- Broadband television distribution service (draft rec. F.821);
- Broadband HDTV (draft rec. F.822).

Some examples of release 1 user services are simple point-to-point calls with a single connection which has an 'unrestricted' bandwidth: Continuous Bit Rate (CBR) connection oriented services with end-to-end timing, Variable Bit Rate (VBR) connectionless services without end-to-end timing, and unrestricted bearers with proprietary ATM Adaptation Layer (AAL) [12]. Release 2 user services are a superset of the release 1 set, with additions like multi-connection call, multipoint calls, and VBR connection oriented services with end-to-end timing. Release 3 adds multimedia and distributive services.

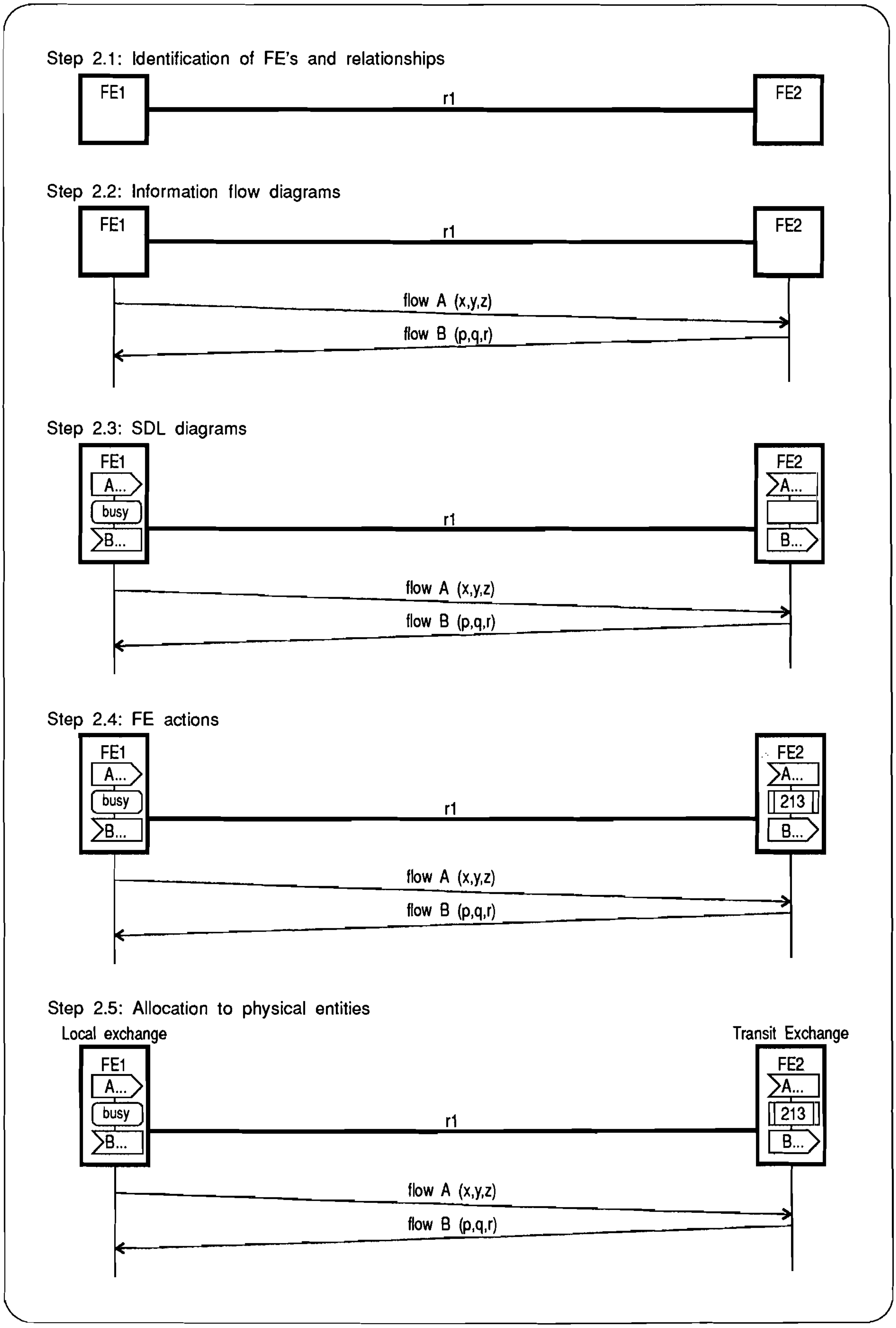


Fig. 2.4 Characterization of functional network aspects.

2.7.2 Stage 2 B-ISDN Signalling

Because the subject of this report is the stage 2 development of B-ISDN signalling releases 2 and 3, the state of the art of this stage will be described more precisely. Involving the basic call, both releases can be grabbed together because release 3 will be a superset of release 2.

At the start of this graduation task, the stage 2 description enclosed a functional model (step 2.1), a number of information flow diagrams (step 2.2), and a part of the overall B-ISDN stage 2 requirements [1,11]. The next chapter will go into Q.73 stage 2 modelling. Yet the functional model is 'frozen' and a possible scenario for allocation of FEs to Physical Entities (PEs) is also identified (step 2.5). The information flow diagrams that already have been developed are of the scenarios:

- Point-to-point: simultaneous establishment of a call with one connection;
- Point-to-point: simultaneous establishment of a call with one connection (overlap sending);
- Point-to-point: sequential establishment of a call with one connection;
- Point-to-point: simultaneous release of a call with one connection;
- Point-to-point: release of a connection within a call;
- Point-to-point: release of a call without connections;
- Point-to-point: simultaneous establishment of a call with multiple connections;
- Point-to-point: simultaneous release of a call with multiple connections.

Point-to-multipoint (p–mp), mp–mp, and mp–p information flow diagrams have not been developed yet. See chapter 3 for details.

2.7.3 Stage 3 B-ISDN Signalling

At this time the development of stage 3 is focused on protocol architecture, protocol developm-ent, and signalling protocol evolution [1,6,7,11]. The main activities are concentrated on release 1. Below some examples are given to show what is going on. Of course this is no complete list of activities. The examples are taken from CCITT contributions which were available at the beginning of this graduation project.
Signalling protocol stacks are being developed based on the OSI principles [14]. Some protocols are in development, like the meta-signalling protocol [8] to set up a signalling virtual channel (VC). Several scenarios of peer-to-peer communication are envisaged of how protocol evolution from release 1 protocol to release 2 protocol could take place by fully reusing the release 1 protocol in a context sensitive manner controlled by the service. Message functional definitions and information content (semantics) are studied and specified [13].

2.8 Conclusions

A control system based on separation between CC and BC and the introduction of a more modular structure results in an improved flexibility which is essential for the second and third release of the broadband ISDN signalling. The new requirements to the signalling system originate from several sources, especially new telecommunication services.
The development of the new signalling system for B-ISDN release 2/3 is now focused on the second and third stage of the three stages methodology for characterization according to CCITT

recommendation I.130. The second stage is divided into five steps that will be passed through several times.

The state of the art of stage 2 comprises a functional model and a number of information flow diagrams for basic point-to-point calls with point-to-point connections. Basic call scenarios involving other call and connection types (p–mp, mp–p, mp–mp) must be developed, as well as scenarios involving other services.

3 Draft Q.73 Functional Models

3.1 Goal of Recommendation Q.73

When finished, CCITT recommendation Q.73 will enclose a stage 2 description of basic call scenarios in the B-ISDN release 2. The draft recommendation is a working document describing the intermediate results. In this chapter some of the major developments are described to show the difficulties encountered so far.

At first a functional model has been stated to be used for point-to-point basic call scenarios. A number of information flow diagram – partly described in this report – are developed for this model.

The next step is the introduction of multiple parties as well as multipoint connection types, especially point-to-multipoint unidirectional connections. To support this type of connections, it is studied whether and how the functional model must be extended. Again a set of information flow diagrams are created based on the modified model. The diagrams form a base for a more stable model and for new information flow diagrams involving other connection types.

3.2 Basic Functional Model

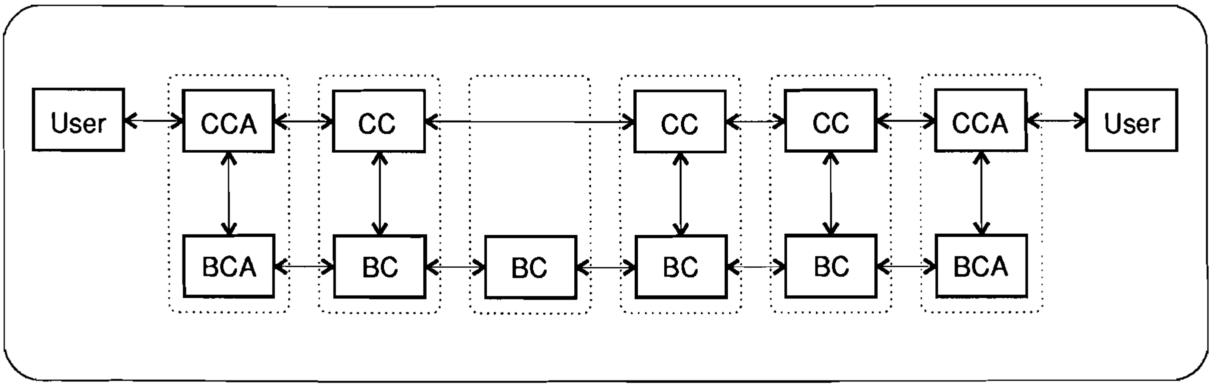


Fig. 3.1 Simple basic functional model.

Figure 3.1 show the functional model which has been the base for draft recommendation Q.73. CC and BC functionalities are separated by using two levels. The model shows that relaying nodes not necessarily have a call control FE. Also can be seen that interaction with the user occurs at the call control level only.

3.2.1 Dynamic Instance of the Model

The next conventions for reference of directions are defined:

- **Originating Local Exchange (LEX)** is at the side of the originating terminal and calling user;
- **Terminating LEX** is at the side of the terminating terminal and called user;
- **Incoming side** of a node is the side closest to the calling user;
- **Outgoing side** of a node is the side closest to the called user;
- **Forward direction** is from the calling user to called user;
- **Backward direction** is from the called user to calling user.

Some FEs are divided into an incoming side and an outgoing side as depicted in figure 3.2. Furthermore the figure shows a possible scenario of allocation of nodes to physical entities. This results in an instance of the functional model as displayed in figure 3.3. For the development of information flow diagrams one of both transit exchanges (TEX) is left out, depending on the need for CC in the relaying node.

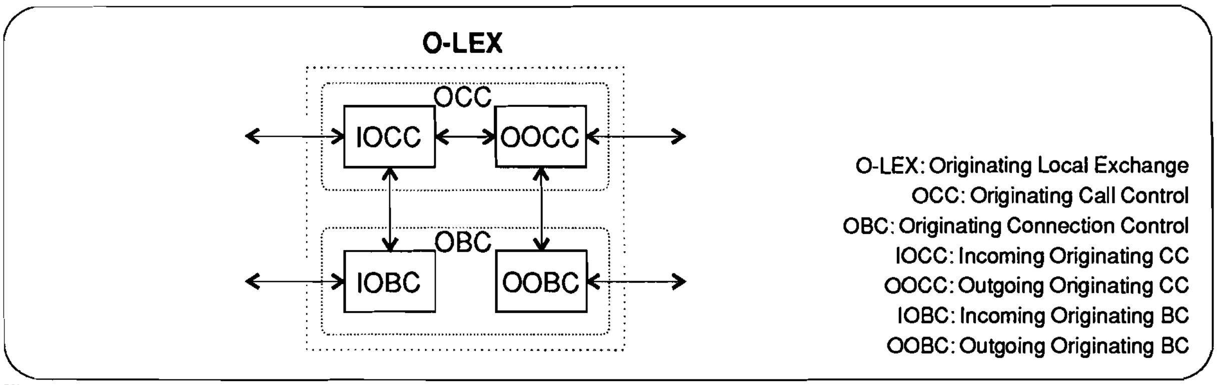

O-LEX
OCC
IOCC
OOCC
OBC
IOBC
OOBC
O-LEX: Originating Local Exchange
OCC: Originating Call Control
OBC: Originating Connection Control
IOCC: Incoming Originating CC
OOCC: Outgoing Originating CC
IOBC: Incoming Originating BC
OOBC: Outgoing Originating BC

Fig. 3.2 The FEs are split into an incoming side and an outgoing side; also the node is allocated to a physical entity: the O-LEX.

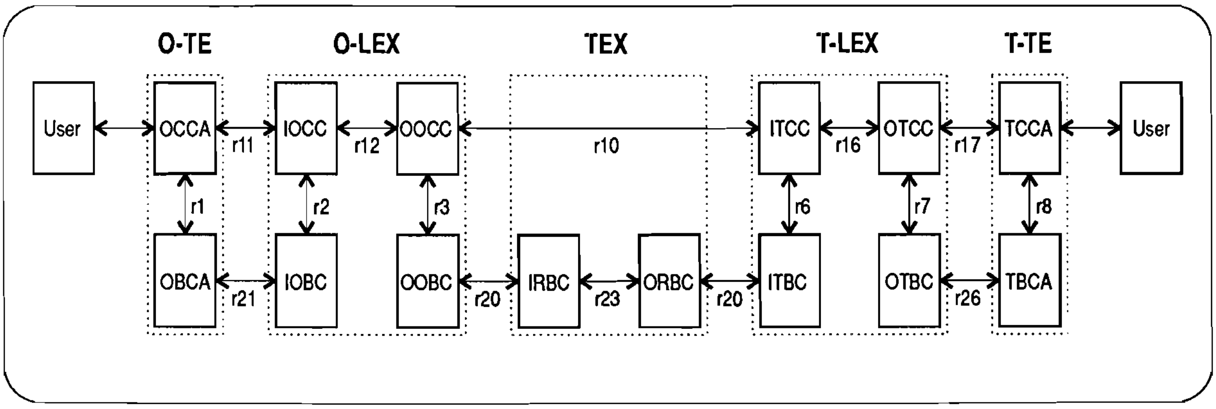


Fig. 3.3 Enhanced functional model without CC in the relaying node.

The functional model depicted in figure 3.3 is used to develop information flow diagrams of basic call scenarios involving two parties. This model suffices for these point-to-point scenarios (see also chapter 4). If a call has more than one connection, several BC instances appear: one instance for the control of each connection (see also section 2.4 and 3.3). In this case the functional model used, is derived from the basic functional model. This model has been adjusted

for one scenario in contrast of the basic functional model: the basic functional model forms the pure and most simple – but complete – model.
The need of a relationship between an incoming BC and an outgoing BC entity within the same node is subject of discussion.

The relationships r12 and r16 may have the same name, for instance r12, because exactly the same information will flow through these relationships. This can be argued as follows:
If two subscribes are connected to the same local exchange all entities between IOCC and OTCC are absent. So only a relationship exists between IOCC and OTCC. The 'left half' of this relationship is called r12, but the 'right half' is called r16. Because by definition no information will be altered when travelling from one side to another through a relationship, must obtain r12=r16 in this particular case. Up to now no differences have appeared in other cases, so r12 and r16 may be identical with an increasing probability.

3.3 Multi-Connection Modelling

The basic functional model has to be modified for scenarios with more than one connection. To explain this, the next example focuses on a call involving two connections (e.g. videophone call). For each BC entity two instances occur. Each instance controls exactly one connection, as can be seen in figure 3.4. Because the model displays only one call, the CC entities appear just once. So each CC entity may control zero or more BC entities. This conforms the requirement of having a call with more than one connection involved in it.

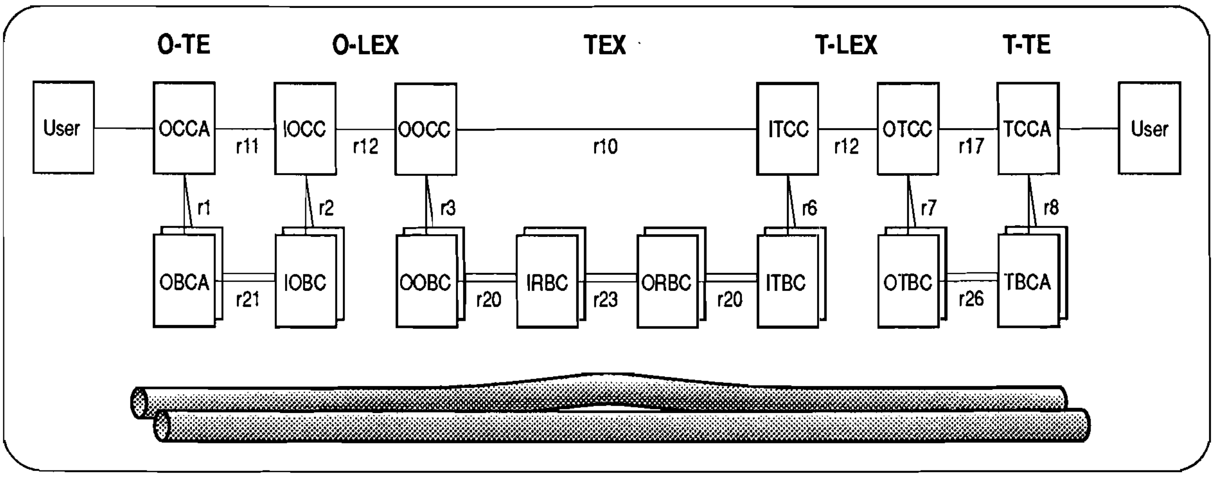


Fig. 3.4 Instance of the functional model for a call with two connections. The tubes below the model depict the connections in the user plane.

3.4 Functional Model for MP Cases

Other basic call scenarios, concerning multipoint cases, introduce the need of a bridge functionality. As has been explained in section 2.2, the existence of a bridge introduces more complex situations. It is not clear yet what level the control of the bridge has to be assigned to. The control of the bridge must not be integrated in the CC level to avoid the need for CC FEs in the relaying nodes. Also it can be assigned to the BC level. In this case the BC entities enclose a number of complex functions. But BC FEs appear many times within a call: one entity for each connection in each node. So this solution may have expensive consequences.

The third option is to introduce another level, called Special Resource Control (RC), situated between the CC level and the BC level. This level controls some of the basic call resources: bridges and connections to these bridges. A RC entity encloses at least the following functions:

- control of the establishment, release, and modification of a connection by instructing the BC level;
- user data distribution (multicast) for P–MP connections;
- user data gathering for MP–P connections;
- combination of gathering, and distribution for MP–MP connections.

In this context the notion "gathering" may be explained in two ways: the user data may be collected and directly mixed, or collected and combined by data processing. The implementation consequences of both explanations differ quite substantial.
The first one implies a concatenation of several incoming channels into one output channel without processing the user data. Only a number of buffers is required. The latter explanation implies the need of fast processing capabilities and standardization of the user data format. This type of bridges requires lots of extra hardware.

The three level functional model is still under discussion. As an international working assumption, it has been agreed that a RC entity will be inserted between every CC and BC entity. So an entire new level has been introduced. In the horizontal direction a relationship appears between all adjacent RC entities. In the vertical direction the direct relationship between CC and BC stays, but two new relationships are introduced for every CC–RC–BC cluster: one between CC and RC and the other between RC and BC. Figure 3.5 shows this complex model.

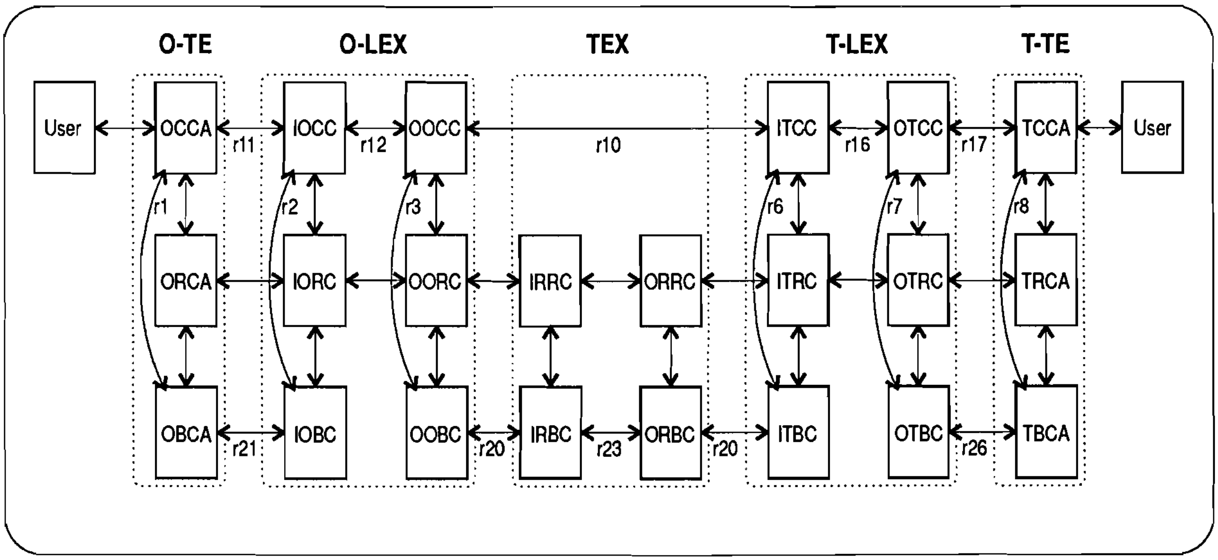


Fig. 3.5 Extended basic functional model having an intermediate level for Special Resource Control.

This model is very hard to deal with. As a working assumption the model has to be simplified as far as possible, without introducing extreme restrictions. At first the following restrictions are assumed:

- The three-level model will be used for MP calls only;

- If a RC entity is present in a node, all information from CC to BC and vice versa will go through the RC entity;
- No RC is required in the TE;
- The relations between OORC–IRRC and ORRC–ITRC are equal.

The first restriction says that the model presented in figure 3.3 will be used for two-party calls. So the old model will not be thrown away. The second restriction preserves the RC entities to be by-passed by the CC entities. Now the RC entities will be informed about all changes in the BC level. The third assumption restricts the allocation of bridges to the internal network. The final restriction can be derived by removing the TEX. In this case OORC will be connected directly to ITRC.

Figure 3.6 shows the three level model that conforms to all mentioned assumptions.

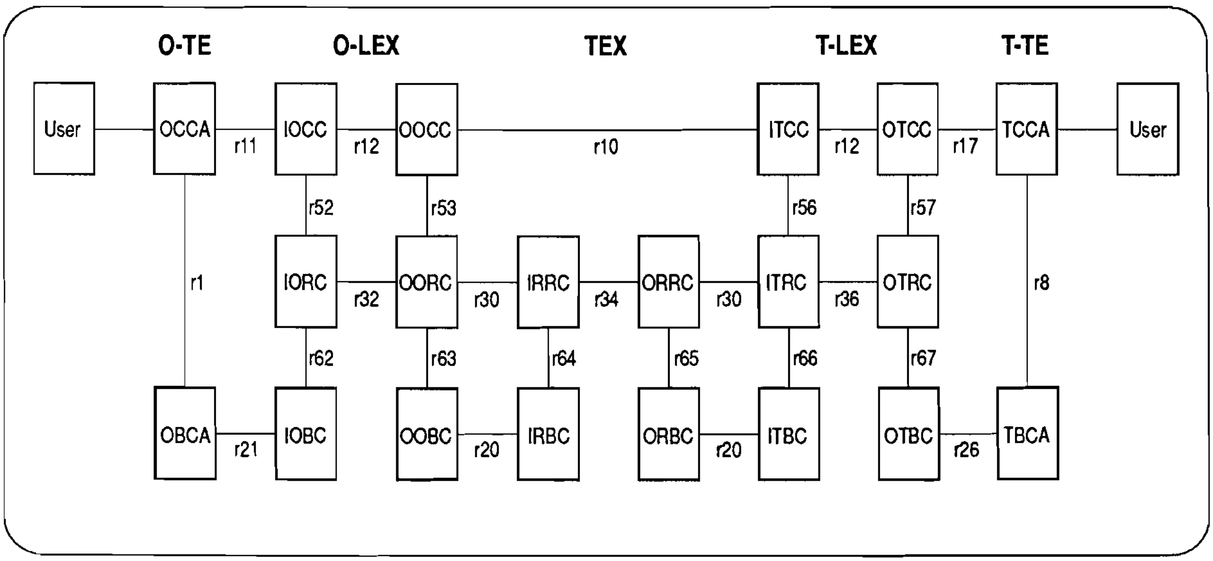


Fig. 3.6 Simplified three level basic model with a possible physical allocation.

3.5 Multi-Party Modelling

The basic functional model has to be modified for scenarios with more than two parties. Figure 3.7 show an example of a three party call with one connection involved. As can be seen, the bridge is located in the TEX. Note the different spots at which the call and the connection are split. The tube below the model depict the connection in the user plane.

3.6 Functionality of FEs

In this section the functionalities of every FE depicted in figure 3.6 are described in detail. A subdivision into the three levels is applied to preserve a structured view. The information described in this section may differ from the standards.

3.6.1 Entities in the BC level

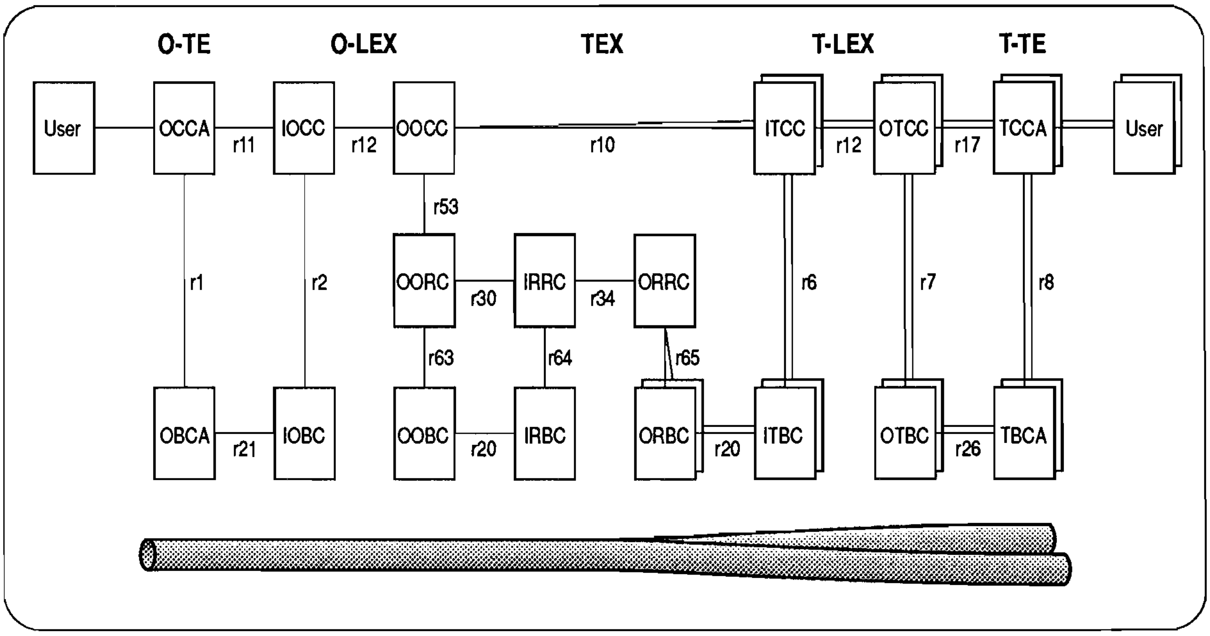


Fig. 3.7 Instance of the functional model for a three party call with one connection. The bridge is located in the TEX. The tubes below the model represent the connection in the user plane.

3.6.1.1 Overview

	OBCA	IOBC	OOBC	IRBC	ORBC	ITBC	OTBC	TBCA
Setup of the association(s) between BC entities:	•	-	•	-	•	-	•	-
Release of the association(s) between BC entities:	•	•	•	•	•	•	•	•
Routing (selection of the next BC node/switching):	-	-	•	-	•	-	•	-
Maintenance of the state of the connection:	•	•	•	•	•	•	•	•
Identification of the connection at the BC level:	•	•	•	•	•	•	•	•
Forward reservation of network resources:	-	•	•	-	•	-	•	-
Backward reservation of network resources:	-	-	-	-	-	-	•	-
Forward allocation of network resources:	-	-	•	-	•	-	•	-
Backward allocation of network resources:	-	•	-	-	-	-	-	-
Forward through-connect:	-	•	-	•	-	•	-	-
Backward through-connect:	-	-	•	-	•	-	•	-
Forward release/deallocation/disconnect:	-	•	•	•	•	•	•	-
Backward release/deallocation/disconnect:	-	•	•	•	•	•	•	-
(Re)negotiation of bandwidth:	•	•	•	•	•	•	•	•
(Re)negotiation of common route connection grouping:	-	-	•	•	•	•	-	-
(Re)negotiation of QOS of bandwidth:	•	•	•	•	•	•	•	•
VCI/VPI negotiation:	-	-	•	•	•	•	-	-
Interworking with existing networks:	-	•	-	•	•	-	•	-
Modification of control rights to BC entities:	•	•	-	-	-	-	•	•
Notification of change in bearer status (progress, connected, ...):	•	•	•	•	•	•	•	•
Policing:	-	•	-	•	•	-	•	-
Delivering charging information:	-	•	-	-	-	-	•	-

3.6.1.2 Details

Setup/release of the association(s) between BC entities. Only outgoing BC entities will set up a new association between itself and the next BC entity. Both incoming and outgoing BC entities are able to initiate a BEARER RELEASE.

Routing (selection of the next BC node/switching). An incoming BC entity needs only to be associated with the next outgoing BC entity. No external link is involved. So only outgoing BC entities require routing capabilities, excluding the OBCA. The OBCA can be associated with an IOBC entity of the one O–LEX.

Note: from a different point of view can be concluded that incoming BC entities must have routing capabilities: An incoming BC entity select a route and creates an instance of an outgoing BC entity to control the selected link.

Maintenance of the state of the connection. Of course each BC entity must be aware of the state of a connection to be able to control it.

Identification of the connection at the BC level. Also each BC entity must be able to identify a connection, otherwise it will be impossible to control one single connection.

Forward reservation of network resources. An outgoing entity reserves resources upon an incoming BEARER SETUP request, except for the OBCA. On the O-UNI the IOBC will take care of the reservation of a forward connection after screening and authorization in the CC level has occurred.

Backward reservation of network resources. Backward reservation functionality is only required in the OTBC entity for the T-UNI. On the O-UNI and the NNI backward reservation is applied implicitly by allocation of a connection.

Forward allocation of network resources. All outgoing BC entities, except OBCA, have to take care for allocation of a forward connection. On the O-UNI forward allocation is applied implicitly by forward through-connection.

Backward allocation of network resources. Backward allocation functionality is only required in the IOBC entity for the O-UNI. On the NNI and the T-UNI backward allocation is applied implicitly by backward through-connection.

Forward through-connect. On all external interfaces the incoming BC entities take care for forward through-connection.

Backward through-connect. On all external interfaces the outgoing BC entities take care for backward through-connection.

(Re)negotiation of bandwidth. All BC entities may be involved in bandwidth negotiation.

(Re)negotiation of common route connection grouping. Common route connection grouping is a feature applicable on network internal links only. So only BC entities involved in NNI interfaces require grouping functions.

(Re)negotiation of QOS of bandwidth. All BC entities may be involved in QOS negotiation.

VCI/VPI negotiation. All BC entities, except on the UNI, require VCI/VPI negotiation functions. The OBCA and TBCA are simply instructed to use a suggested identifier. So no negotiation will be applied with the IOBC and OTBC entities.

Interworking with existing networks. The OBCA and TBCA are assumed to have no interworking functionality, so functions for interworking at the UNI must be present in the IOBC and OTBC. Of course all relaying nodes must be able to interwork with other broadband networks. The OOBC and ITBC don't require any interworking functionality, because interworking will be taken care of by a gate-way in a relaying node.

Modification of control rights to BC entities. Modification of control rights corresponds to changing the ownership of a call. Changes to the control rights of entities is only relevant on the UNI, e.g. for control of policing functions.

Notification of change in bearer status (progress, connected, failure, ...). All BC entities have to be aware of changes and are thus involved in the notification process.

Policing. Policing must be applied at the UNI by the network and in relaying nodes when interworking is utilized with other networks.

Delivering charging information. All needed charging information can be delivered by the IOBC and the OTBC entities. For P–P type connections charging delivery functions are required in the IOBC only.

3.6.2 Entities in the CC level

3.6.2.1 Overview

	OCCA	IOCC	OOCC	ITCC	OTCC	TCCA
Setup of the association(s) between CC entities:	•	-	•	-	•	-
Release of the association(s) between CC entities:	•	•	•	•	•	•
Selection of the next CC node:	-	-	•	-	•	-
Identification of association(s) between CC entities:	•	•	•	•	•	•
Maintenance of the state of the call:	•	•	•	•	•	•
Telecommunication service screening:	-	•	-	-	•	-
User authentication and authorization:	•	•	-	-	-	-
Service component negotiation:	•	•	-	-	•	•
User status checking and monitoring:	•	•	-	-	•	•
(Re)negotiation of end-to-end QOS:	•	•	-	-	•	•
Compatibility checking:	•	•	-	-	•	•
Modification of control rights to CC entities:	•	•	-	-	•	•
Interworking with existing networks (see section 3.6.2.2):	-	•	-	-	•	-
Give order to set up connections:	•	-	•	-	•	-
Give order to release connections:	•	•	•	•	•	•
Coordination between resources:	•	•	•	•	•	•
Identification of connections at CC level:	•	•	•	•	•	•
Collection of charging information:	-	•	-	-	-	-
Give order to set up a bridge/branching node:	-	-	•	-	-	-
Give order to release a bridge/branching node:	-	-	•	-	-	-

3.6.2.2 Details

Setup/release of the association(s) between CC entities. Only outgoing CC entities will set up a new association between itself and the next CC entity. Both incoming and outgoing CC entities are able to release a call.

Selection of the next CC node. An incoming CC entity needs only to be associated with the next outgoing CC entity. No external relationship is involved. So only outgoing CC entities require node selection capabilities, excluding the OCCA. The OCCA can be associated with an IOCC entity of the one O–LEX.

Identification of association(s) between CC entities. Of course each CC entity must be able to identify an association.

Maintenance of the state of the call. Also each CC entity must be aware of the state of the call to be able to control it.

Telecommunication service screening. Screening is only needed at the O–UNI and the T–UNI. Screening at the T–UNI is necessary to check if the called user is also qualified to use or subscribed to a service.

E.g. consider a two party call. The B user takes over the ownership of the call. So if no screening is applied at the T–UNI, the B user can use a service without subscription.

User authentication and authorization. The authentication and authorization process are required at the O–UNI but not at the T–UNI, because this process may be fulfilled without setting up any call.

Service component negotiation. Service component negotiation is an end-to-end matter, so all CC entities at the UNIs require functions.

User status checking and monitoring. Of course user status is only relevant at both UNIs, so OOCC and ITCC entities don't need functions to support user status checking and monitoring.

(Re)negotiation of end-to-end QOS. Both the network side and the terminal side of the UNIs are involved in end-to-end QOS negotiation.

Compatibility checking. Compatibility checking applied by IOCC and OTCC entities to find out the terminal type. Also the terminals have to do compatibility checking to find out if it is connected to the right network type.

Modification of control rights to CC entities. Modification of control rights corresponds to changing the ownership of a call. Changes to the control rights of entities is only relevant on the UNI.

Interworking with existing networks. The OCCA and TCCA are assumed to have no interworking functionality, so functions for interworking at the UNI must be present in the IOCC and OTCC. The OOCC and ITCC don't require any interworking functionality, because a LEX is by definition of a different network type if interworking is applied by the OOCC or ITCC. If interworking is required in an relaying node, this node will require (R)CC functionality that is capable to interwork with an existing network. So IRCC and ORCC entities require – if present – interworking functions.

Give order to set up/release connections. All outgoing CC entities may give order to the BC or RC level to set up a connection. Both incoming and outgoing CC entities may give order to the BC or RC level to initiate a BEARER RELEASE.

Coordination between resources. All CC entities have supervision of the call and the resources involved. So all entities in the CC level must be able to coordinate resources.

Identification of connections at CC level. Of course all CC entities must be able to identify a connection in order to be able to control it. If a RC entity occurs between a CC and a BC entity, the RC entity will also take care of the identification.

Collection of charging information. Only the IOCC requires charging information gathering functions assuming the O-user is the owner of the call.

Give order to set up a bridge/branching node. If a bridge or a branching node are involved with a call, the initial instantiation of the RC level takes place in the O–LEX. The OOCC has to give order to the OORC entity to setup a bridge or one or more branching nodes.

Give order to release a bridge/branching node. The OOCC can give order to release bridges or branching nodes. E.g. a T–user may hang up the phone and thus give order to drop a party/branch. If two parties are left a bridge may be released and a branching nodes stops branching on user request. But in a two party call the owner may prefer to keep the bridge, even if the third party hangs up.

3.6.3 Entities in the RC level

3.6.3.1 Overview

	IORC	OORC	IRRC	ORRC	ITRC	OTRC
Setup of the association(s) between RC entities:	-	•	-	•	-	-
Release of the association(s) between RC entities:	-	•	•	•	•	-
Selection of the next bridge:	-	•	-	-	-	-
Selection of the next branching node:	-	•	-	•	-	-
Identification of association(s) between RC entities:	•	•	•	•	•	•
Maintenance of the state of a bridge:	•	•	•	•	•	•
Identification of a bridge/branching node at the RC level:	•	•	•	•	•	•
Bridge hand over:	-	•	•	•	•	-
Add party/branch:	•	•	•	•	•	•
Drop party/branch:	•	•	•	•	•	•
Notification of bridge progress:	•	•	•	•	•	•
Modification of control rights to RC entities:	-	-	-	-	-	-

3.6.3.2 Details

Setup/release of the association(s) between RC entities. Only outgoing RC entities at the NNIs will set up a new association between itself and the next RC entity. The OTRC has no next entity, so it needs no association setup functions. Both incoming and outgoing RC entities at the NNIs are able to release a association. The IORC and OTRC entities have no external relationships, so they don't need association release functions.

Selection of the next bridge. Only the OORC requires bridge selection functionality, assuming bridges to be non-distributed. If a bridge setup fails, the OORC selects a new one. When a bridge is allocated to the O-LEX or T-LEX no selection functions are required.

Selection of the next branching node. Both the OORC and the ORRC entities require branching node selection functions, because branching may be distributed in the network.

Identification of association(s) between RC entities. Of course each RC entity must be able to identify an association.

Maintenance of the state of a bridge. Also each RC entity must be aware of the state of a bridge to be able to control it.

Identification of a bridge/branching node at the RC level. Of course all RC entities must be able to identify a bridge or branching node.

Bridge hand over. Only internal RC entities require hand over functionality.

Add party/branch. All RC entities require addition functions because a new branch may be added in the bridge or in a branching node wherever that bridge or node appears.

Drop party/branch. Of course all RC entities may drop a party or a branch.

Notification of bridge progress. Each RC entity must be aware of the status of the bridge and are thus involved in the notification process.

Modification of control rights to RC entities. Control rights of RC entities cannot be changes. When the owner of the call changes, hand over will take place in the RC level.

3.7 New Developments

The models described in this chapter are not stabilized yet. Especially the three level model is very generic and needs some time to crystallize or to be dropped. Several new developments can be observed, based on three statements:

- Some simplifications are agreed because the time limit of December 1993 for stage 2 of B-ISDN release 2 must be met;
- B-ISDN release 2 and 3 should evolve more directly from B-ISDN release 1;
- Modifications and new concepts as a result of research.

As a simplification it is agreed that MP–P[1] and MP–MP connections will not be supported by release 2. In other words, only bidirectional P–P and unidirectional P–MP type connections will be modelled. So the connections meet exactly definition 2 of figure 2.1, which is also a working assumption for this report.
More direct evolution finds expression in several models, including a monolithic model as used in Q.71 [2]. Following two examples of two-level models illustrate some proposed modifications:

1. The lower level should have the same functionality as Q.71 (consequently called CC level) and the higher level should contain all extra functionality;
2. The lower level used for link-by-link control and the higher level for end-to-end control.

These proposals are partially introduced by developers of stage 3.
Some minor modifications are proposed as a result of intended major changes of the stage 3 protocol stack. A new concept is the use of object-oriented descriptions of entities (and entire nodes). The model proposed for this concept is depicted in figure 3.8. The AC (Application Coordination) entities represent objects which contain shared variables, tables and functions. The CC and BC entities interact with the environment and use the AC objects by manipulating data and calling any functions. A drawback of the AC model is the extra internal relationships compared to the two level functional model.

3.8 Conclusions

Several functional models are proposed for recommendation Q.73. The two level model as depicted in figures 3.1 and 3.3 is already very stable. For the description of multiparty scenarios this model is too restricted, so new - more complex - models are proposed. Up to now the very complex three level model (CC, RC, and BC levels) is accepted for MP cases. In the near future the three level model may be rejected, because it is too complex. The model shown in figure 3.8 has a real change to be accepted as the model to describe both point-to-point scenarios as well as point-to-multipoint scenarios. The last model settles with questions like which FE on what level must take care of through-connection, allocation, and reservation.

The functionality of every FE is identified, though the list of functions may be incomplete. More important is the following: If the RC level will be rejected for some reason, the functions identified with the RC level must be rearranged in the CC and BC levels.

[1] ETSI still supports MP–P connections.

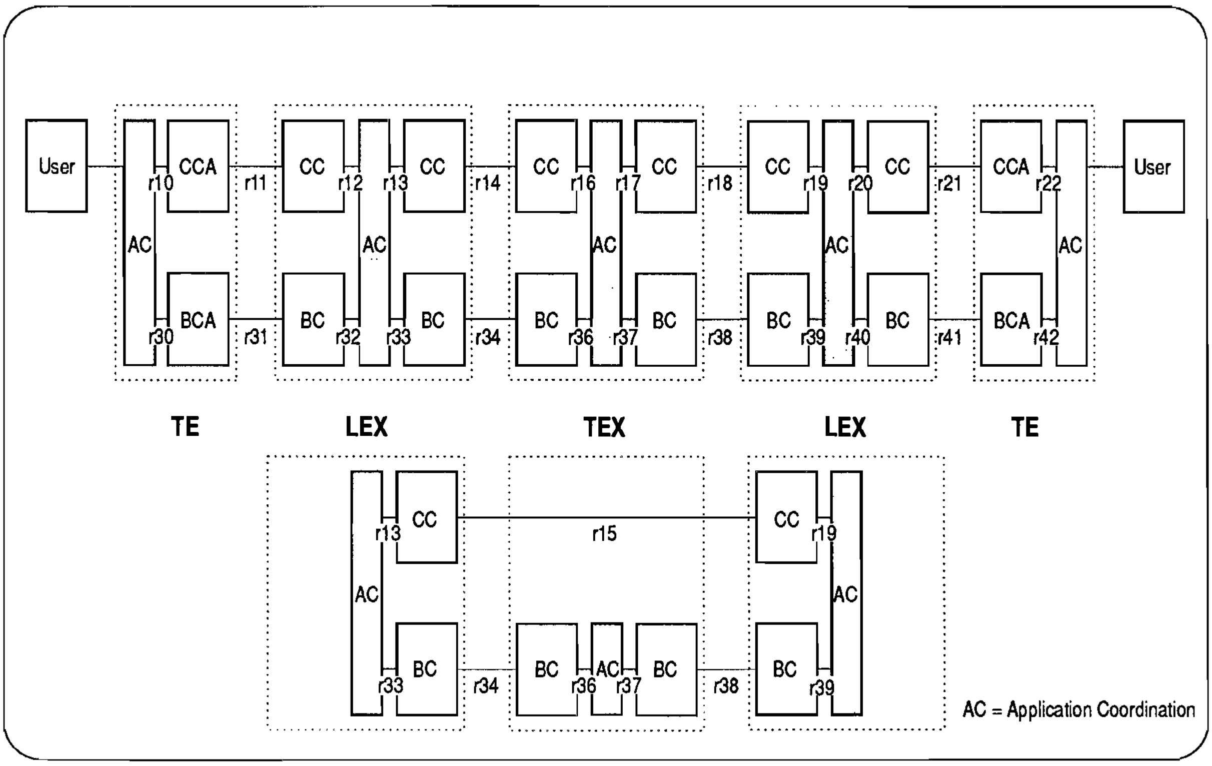


Fig. 3.8 Proposed model suited for object-oriented FE description.

4 Information Flow Diagrams

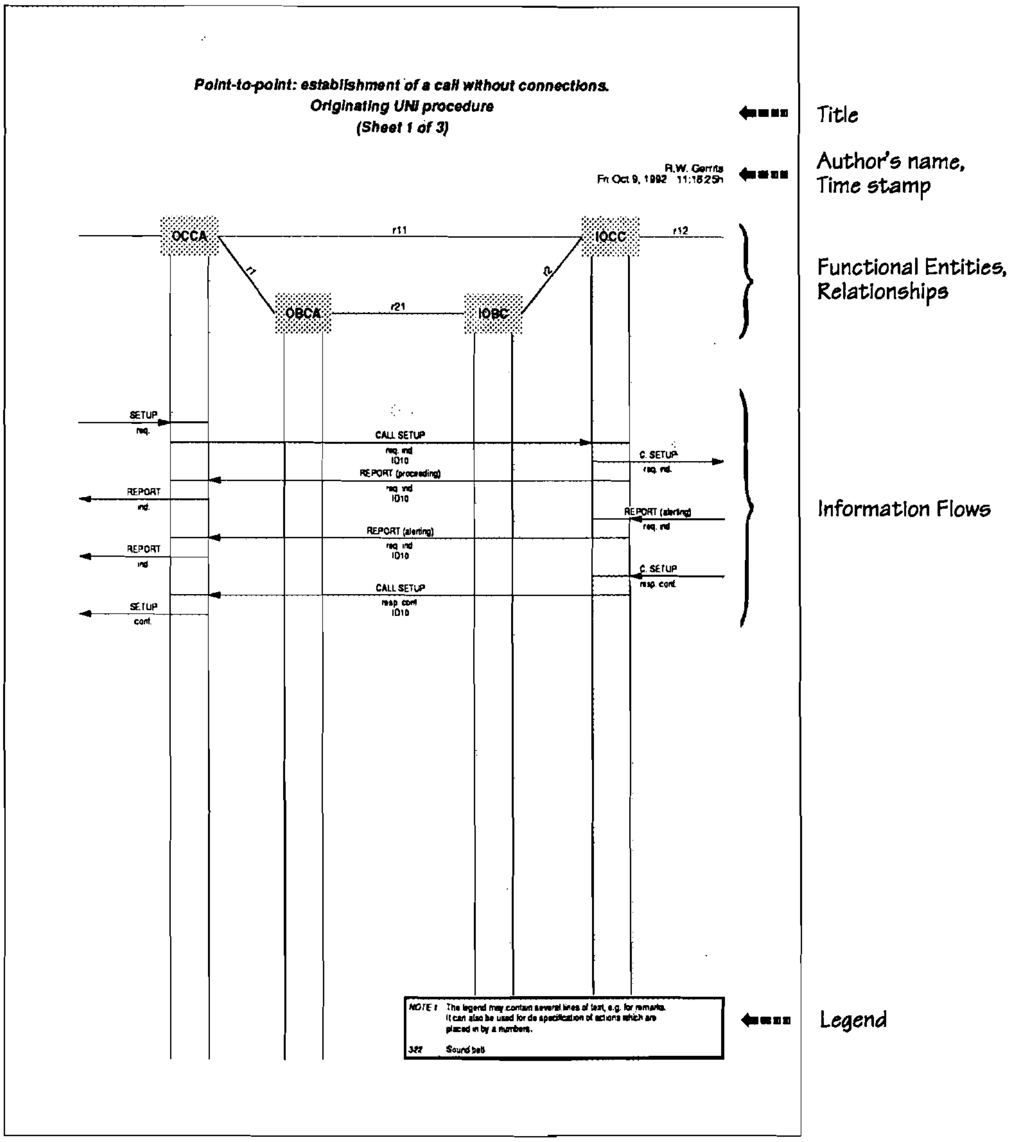
4.1 Introduction

An information flow diagram describes which information must be interchanged between FEs for a given scenario. In this section will be explained how to read an information flow diagram. A diagram may be divided into several pages. Figure 4.1 shows an example of a simple page. Two important areas can be identified:

- Functional entities and relationships
- Information flows and actions

All FEs and relationships together form an instance of the basic functional model. FEs positioned on top of each other in the model are placed next to each other in the information flow diagram. This is necessary to identify the start and endpoint of an information flow. The left margin of the first page and the right margin of the last page represent the users. A user may be a human user or some equipment. If a FE has more than one instance these instances are depicted on top of each other, as can be seen in figure 4.2.
The second area must be read from top to bottom (the time axis). In this area an information flow is depicted as an arrow with several labels. The labels describe the type of information transferred from the source FE to the destination FE. Optionally some ID numbers are added. These numbers describe information elements which must be present in the information flow. The ID numbers are described in the baseline text for recommendation Q.73 [16].

Actions executed by an FE upon the arrival or departure of an information flow are depicted in two ways: by using labels and by using icons. An action label is a code (e.g. a integer number) that describes the action which takes places in a FE. The action labels are placed within the vertical lines under the FEs.
Action icons describe a subset of all possible actions involving the user plane: reservation, allocation, through-connection, and release of bandwidth. The icons are enumerated in figure 4.3. The three icons describing states only are not standardized yet.
The icons happen to be confusing, because sometimes they rather describe a state than an action. In this report the icons are used to describe a state, which is also a tendency within the standardization bodies. However appendix B uses icons to describe actions, unless otherwise mentioned.



Title

Author's name,
Time stamp

Functional Entities,
Relationships

Information Flows

Legend

Fig. 4.1 Example of the page lay out.

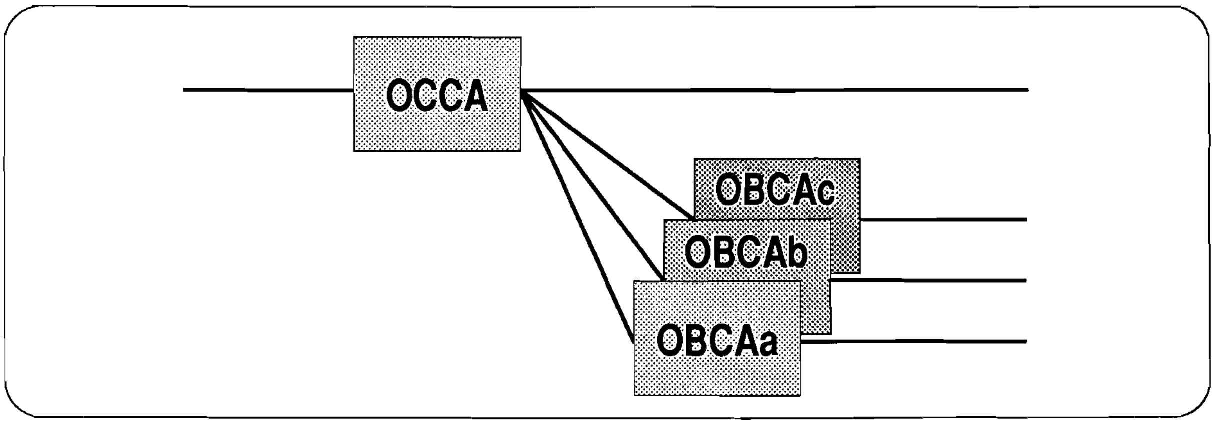
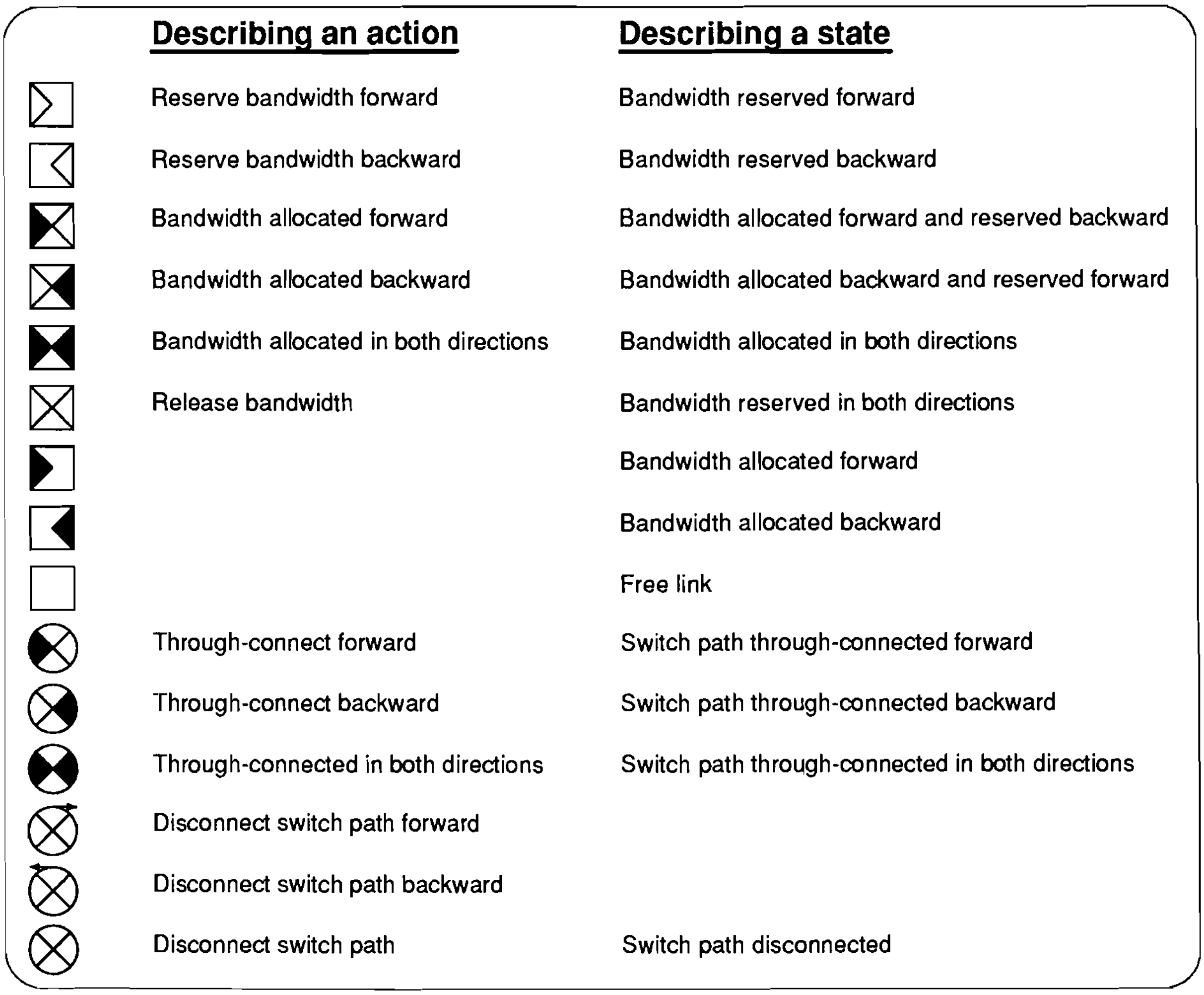


Fig. 4.2 One functional entity may have several instances.

This chapter will go into a selection of the scenarios that I have developed. At first some point-to-point scenarios are described and some problems and decisions are introduced. Next point-to-multipoint scenarios with branching nodes will be discussed. Finally the most complex point-to-multipoint scenarios with a bridge are described.



	Describing an action	Describing a state
▷	Reserve bandwidth forward	Bandwidth reserved forward
◁	Reserve bandwidth backward	Bandwidth reserved backward
⊠	Bandwidth allocated forward	Bandwidth allocated forward and reserved backward
⊠	Bandwidth allocated backward	Bandwidth allocated backward and reserved forward
⧓	Bandwidth allocated in both directions	Bandwidth allocated in both directions
⊠	Release bandwidth	Bandwidth reserved in both directions
▶		Bandwidth allocated forward
◀		Bandwidth allocated backward
□		Free link
⊗	Through-connect forward	Switch path through-connected forward
⊗	Through-connect backward	Switch path through-connected backward
⊗	Through-connected in both directions	Switch path through-connected in both directions
⊗	Disconnect switch path forward	
⊗	Disconnect switch path backward	
⊗	Disconnect switch path	Switch path disconnected

Fig. 4.3 Overview of all possible icons.

Section 4.5 goes into scenarios based on the proposed model with AC entities (see also figure 3.8) and section 4.6 deals with the individual information flows.

4.2 Point-to-Point Information Flow Diagrams

In this section some point-to-point basic call scenarios are described, like successful cases of different types of call and connection setup and release.

A simple two party basic call can be set up in several ways:

- Set up the call only (and add connections in a later stage);
- Set up the call with the connection(s) simultaneously;
- Sequential setup of the call and the connection(s);
- Set up the call using overlap sending.

These different types will be discussed in section 4.2.5. First some simple scenarios will be presented to explain the basic principles and problems.

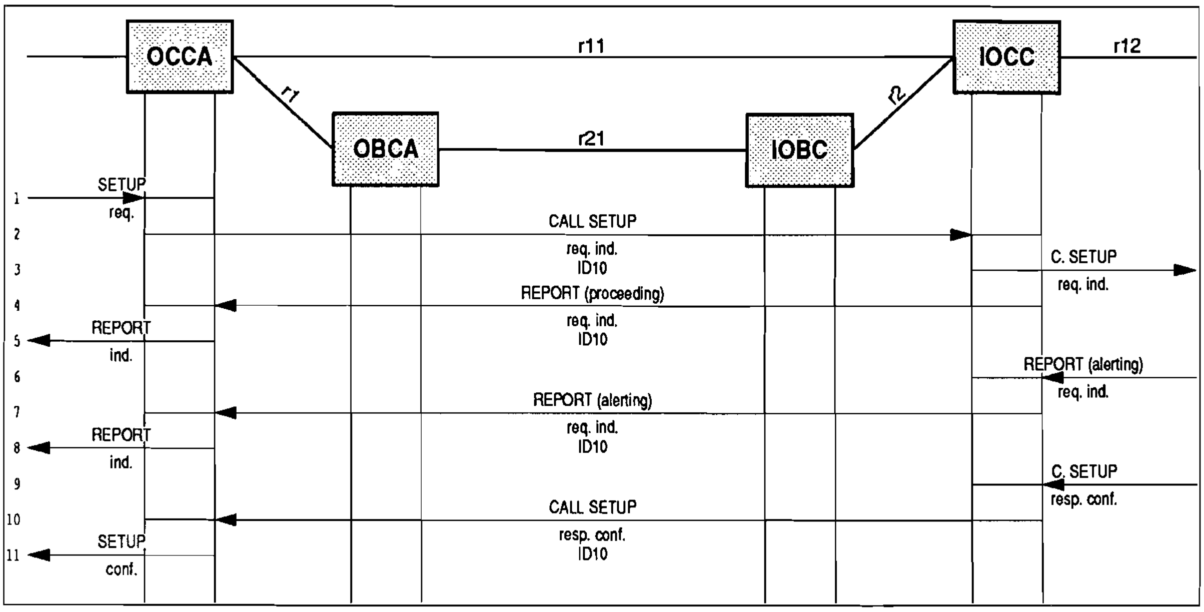


Fig. 4.4a Set up of a call only (O-UNI procedure).

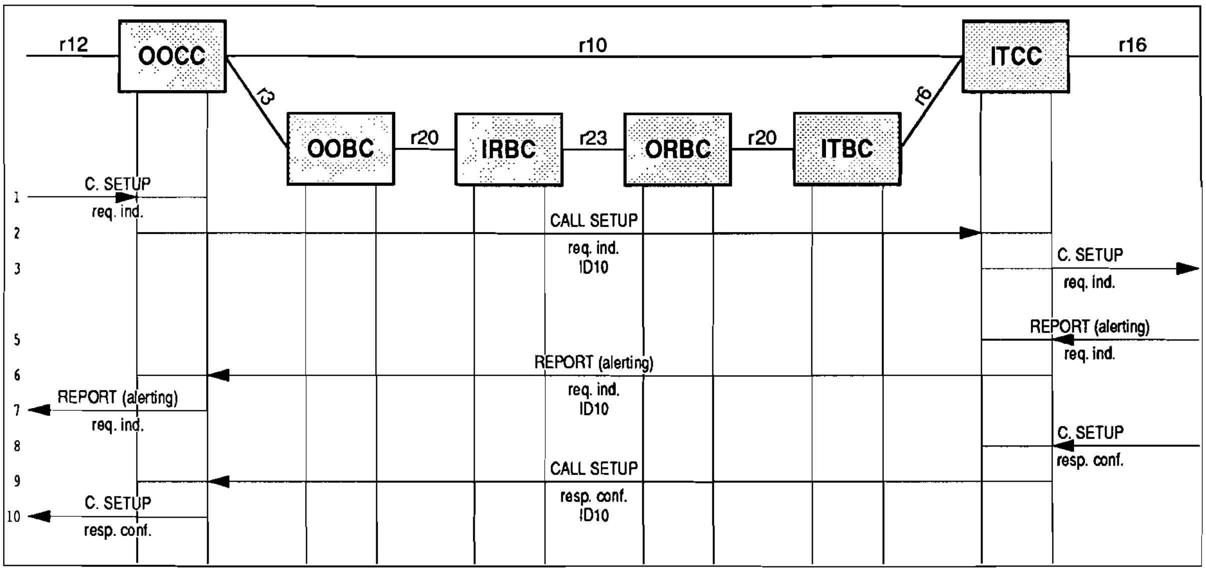


Fig. 4.4b Set up of a call only (NNI procedure).

4.2.1 Set Up Call Only

The information flow diagram depicted in figure 4.4 shows a setup scenario of a call without connections. In fact the three parts of the diagram must be placed next to each other. Although BC entities are not used in this scenario, they are depicted so the model can be better recognized.

In line 1 of figure 4.4a the originating user initiates a call by sending a **SETUP req.** information flow to the OCCA, e.g. by picking up the phone and dialling a number. The terminal equipment sends a **CALL SETUP req. ind.** to the IOCC as a reaction to this information flow. The IOCC forwards the information flow to the OOCC by means of a **C. SETUP req. ind.** flow and sends back a **REPORT (proceeding) req. ind.** flow to inform the OCCA that it has received and processed the setup flow. The OCCA reports the user (e.g. by generating a tone in case of a normal speech phone call) the new state. The proceeding flow can also be used to stop a timer which was started after sending the setup flow. If the timer expires before any response returns from the exchange some action can be taken, like re-sending the last information flow or informing the user that the exchange has

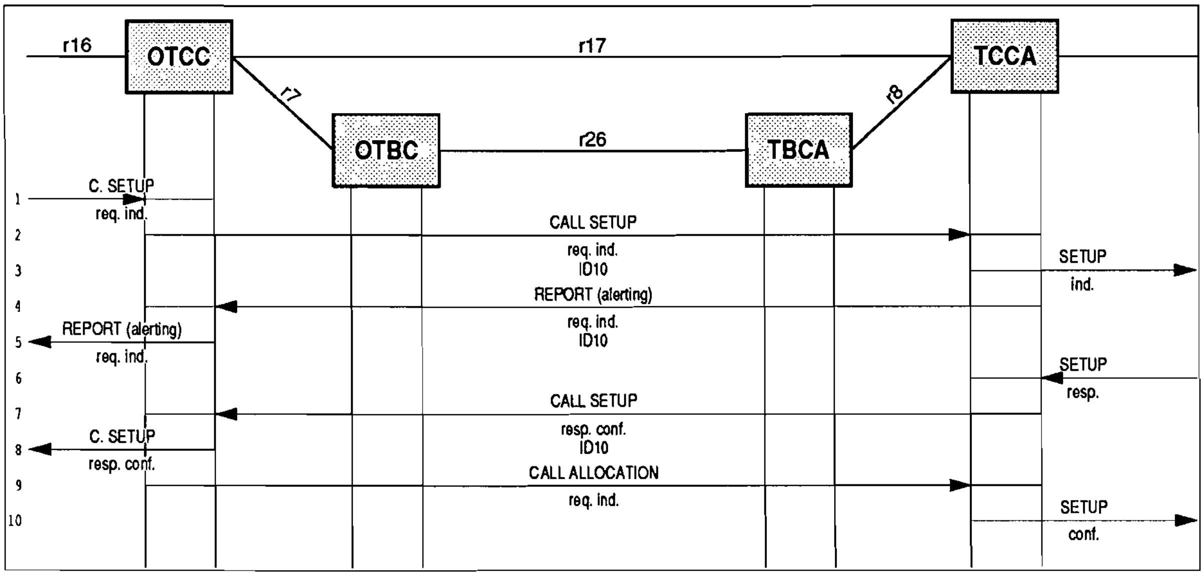


Fig. 4.4c Set up of a call only (T-UNI procedure).

failed.

When **CALL SETUP req. ind.** is received by the TCCA in figure 4.4c, the called user will be informed about the incoming call by a **CALL ind.** flow. Also a **REPORT (alerting) req. ind.** information flow is sent back to the originating CCA. The calling user can now be informed that the call has reached the called party.

After a certain delay the called party reacts by a **SETUP resp.** flow (e.g. by picking up the phone). The **CALL SETUP resp. conf.** information flow will be send through the network to the originating user to inform that the setup has been completed. Also the OTCC sends a **CALL ALLOCATION req. ind.** to the responding terminal. If more terminals have responded to the **CALL SETUP req. ind.** flow, this flow will allocate the call to the first responding terminal and inform other terminals to stop alerting.

ID10 is an information item which identifies the association between two adjacent CC entities.

In case of human users a call without connections doesn't make much sense, because you cannot chatter much without any connections. But if both parties are computers, it is no problem to set up the call first and connections afterward. Depending on the service, alerting the user may be delayed until at least one connections has been set up.

4.2.2 Add a Connection to an Existing Call

If a user wants to add a connection to an existing call (e.g. add a video connection for videophony) the OCCA receives an **ADD BEARER req.** information flow, as can be seen in figure B10-6a of appendix B. The OCCA informs the IOCC with an **ADD BEARER req. ind.** information flow and sends a **BEARER ENQUIRY req. ind.** to the originating BCA. The local exchange knows that a bearer will be set up, so the IOCC waits for a **BEARER ESTABLISHMENT req. ind.** flow. This flow notifies the establishment of a relation between the OBCA and IOBC entities.

Thus the CC level instructs the BC level to set up a connection. In the CC level information has been sent from OCCA to IOCC about changes to take place. The IOCC will inform the OOCC in an equivalent way. In the BC level the bearer will be set up on CC request. The CC will be informed about any progress, in order to control the BC level.

ID2 is an information item which identifies the association between two adjacent BC entities. **ID1** identifies the "bearer" between two adjacent BC entities [16, pp 41-44].

In figure B10-6 of appendix B action icons are depicted near most BC FEs. Following strategies are used for establishment or release of connections:

- Connections in the forward direction are reserved upon forward travelling **BEARER SETUP req. ind.** and **BEARER ENQ. req. ind.** flows. Both allocation and through-connection of forward directed connections takes place upon backward travelling **BEARER SETUP resp. conf.** and **BEARER EST. resp. conf.** information flows. This ensures that no user data can be sent until the entire bearer has been set up and charging has started;
- Connections in the backward direction are allocated and through-connected as soon as possible (upon **BEARER SETUP req. ind.** and **BEARER ENQ. req. ind.**), excluding the terminating UNI. So the T-user is unable to send user data until the bearer is explicitly allocated by a **BEARER ALLOCATION req. ind.** information flow.
- By definition the BCA FEs do not reserve, allocate, or through-connect any connection. They simply accept or reject a setup request.

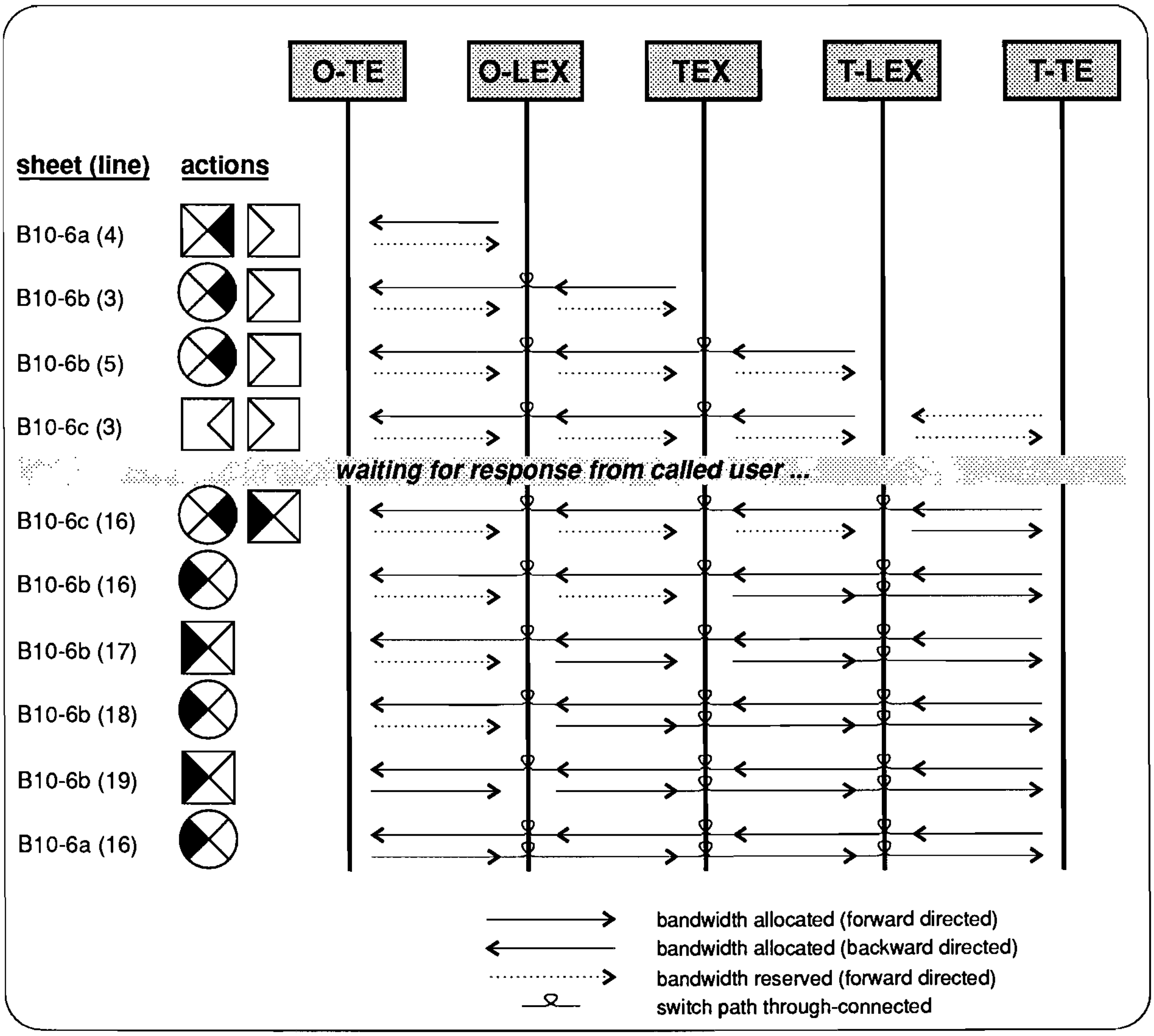

O-TE
O-LEX
TEX
T-LEX
T-TE
sheet (line)
actions
B10-6a (4)
B10-6b (3)
B10-6b (5)
B10-6c (3)
waiting for response from called user ...
B10-6c (16)
B10-6b (16)
B10-6b (17)
B10-6b (18)
B10-6b (19)
B10-6a (16)
bandwidth allocated (forward directed)
bandwidth allocated (backward directed)
bandwidth reserved (forward directed)
switch path through-connected

Fig. 4.5 Add a connection to an existing call: activity in the user plane.

In the light of the above mentioned strategies, the action icons in figure B10-6 are easier to understand. Sometimes it is not obvious which external link is involved. Figure 4.5 illustrates the activity in the user plane for this scenario.

Observe the action icons depicted in figure B10-6b on lines 17 and 19. These actions do not apply on the expected external relationships between ORBC – ITBC and OOBC – IRBC. Figure 4.5 shows that these connections are already allocated.

Through-connection of bandwidth implies allocation if no bandwidth has been allocated before. The upper half of figure 4.5 illustrates this implied allocation of bandwidth twice. Section 4.2.4 will go deeper into bandwidth allocation.

4.2.3 Release of a Call With One Connection

Figure B10-7 in appendix B shows the information flow diagram of the release of a call with one connection. If the user sends a **RELEASE req.** flow to the OCCA, a **CALL RELEASE req. ind.** information flow is sent in the CC level to the IOCC. When finished, the IOCC send a **CALL RELEASE resp. conf.** flow to confirm that the call with it's connections has been released. In the BC level a comparable set of information flows are transferred.

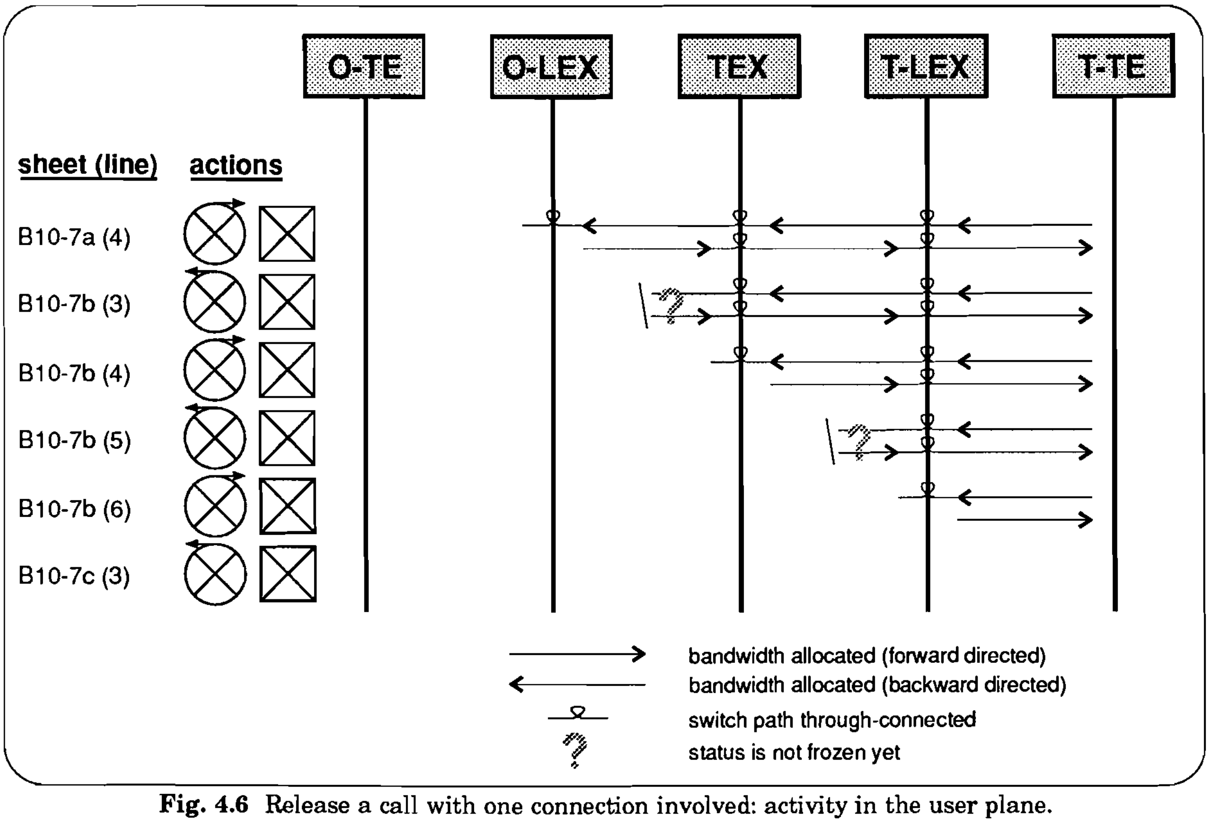


Fig. 4.6 Release a call with one connection involved: activity in the user plane.

Figure 4.6 shows the activity in the user plane. The release of bandwidth and disconnection actions are used in such a way, that some revision will certainly be needed: Bandwidth will be released even *before* the connections have been disconnected. In the forward direction this is allowed, but in the backward direction this strategy will cause troubles: The next adjacent node will still transmit user data backward, but the receiving node doesn't know what to do with the user data transmitted via an undefined connection.

Section 4.2.4 will discuss a better procedure. Of course another solution exists: the way B-ISDN release 1 and N-ISDN take care of bandwidth deallocation [2].

4.2.4 Bandwidth Allocation and De-Allocation

4.2.4.1 Bearer Setup

Both the procedures for allocating and de-allocating bandwidth, including through-connection respectively disconnection, are currently still incomplete.

When a connection is set up one BC entity is always the "assigning node" and the next adjacent BC entity is the "non-assigning node." Only the assigning node is able to select the required resources on a particular link. In practice the outgoing BC FE may either allocate or just reserve bandwidth on a bearer setup request information flow.

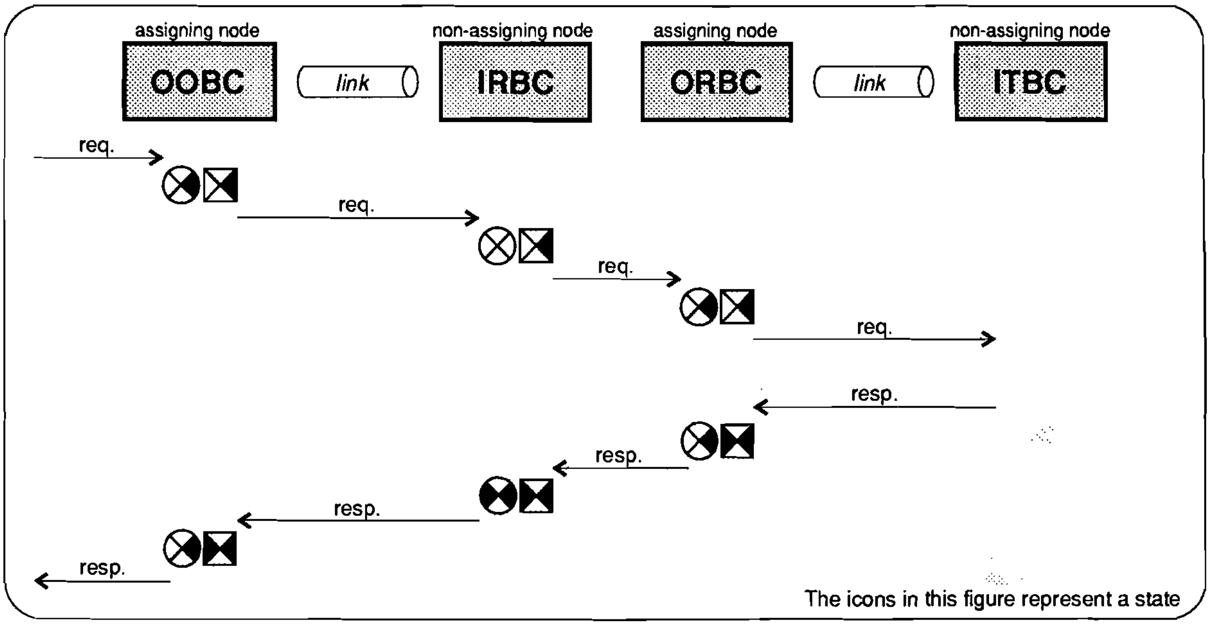


Fig. 4.7a Generic NNI bearer setup procedure with left-to-right assignment priority decrease.

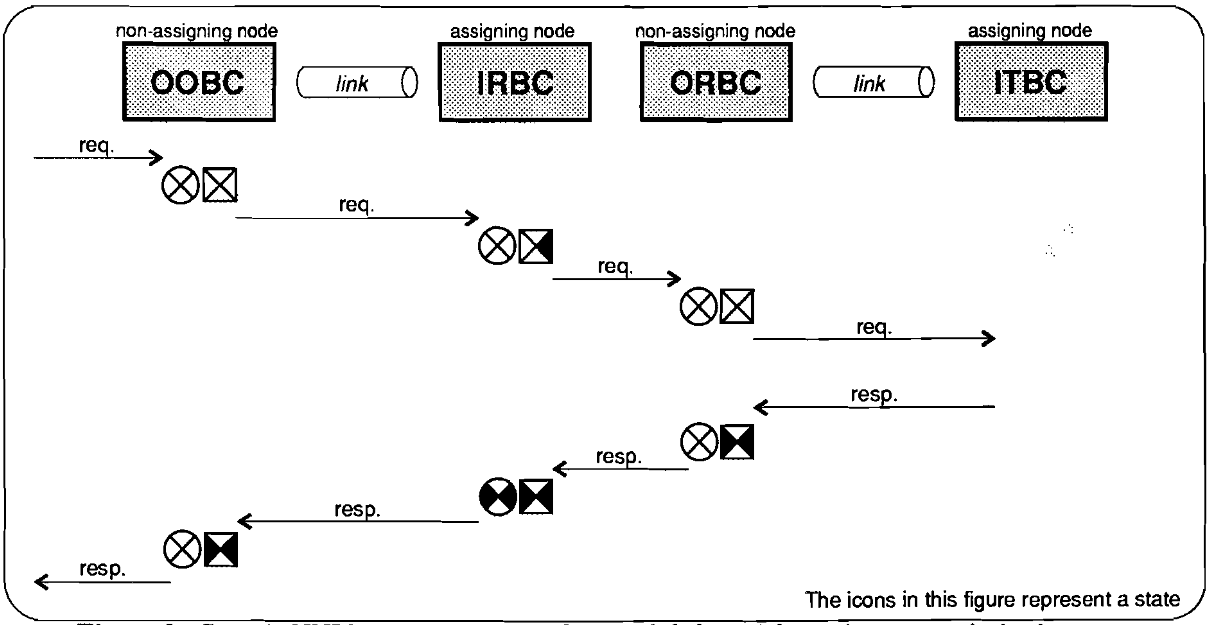


Fig. 4.7b Generic NNI bearer setup procedure with left-to-right assignment priority increase.

Another underexposed item in the currently existing information flow diagrams is the fact that bandwidth has to be allocated *on both sides* of a link. The information flow diagrams in the baseline text [16] sometimes use an ambiguous allocation procedure. Bandwidth allocation actions may be explicitly indicated or assumed to occur implicitly by through-connection actions.

The information flow diagrams in figure 4.7 show two NNI procedures to reserve, allocate and through-connect a bidirectional bearer. In figure 4.7a the assignment priority of the outgoing BC FEs is higher than the assignment priority of the next adjacent incoming BC entity. So the outgoing BC FE controls the link. In figure 4.7b the opposite it true. Of course combinations of figure a and b will appear also.

Conclusions: The action that have to be performed upon the arrival of an information flow depends on the (locally defined) priority of two adjacent nodes. An outgoing BC FE will at least reserve bandwidth and may allocate bandwidth upon the arrival of a requesting information flow. The actions that a FE has to take to through-connect the switch path depends on the priority on both sides of the node.
The reservation and allocation icons apply on a single link, so both incoming and outgoing BC entities must reserve and/or allocate bandwidth. The through-connection icons apply on the entire node.

4.2.4.2 Bearer Release

Section 4.2.3 explained already that the proposed bearer release procedures are incorrect. The information flow diagram in figure 4.8 shows a proper NNI bearer release procedure. Backward disconnection will happen solely after appropriate bandwidth de-allocation has occurred. No responding information flow is sent from ORBC to IRBC, corresponding to the internode information flows in the CC level (see also figure B10-9 of appendix B).

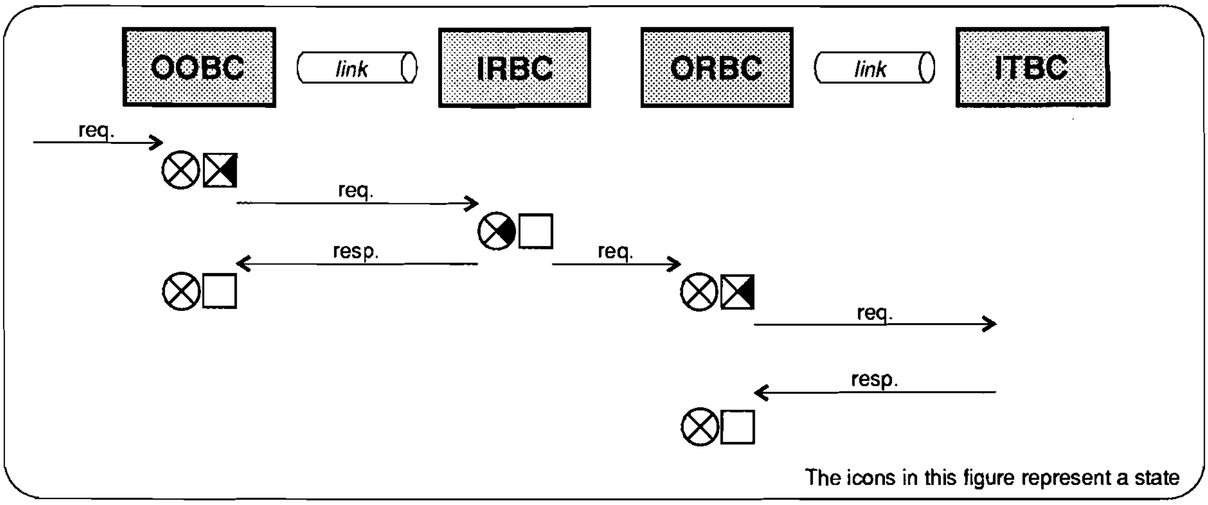


Fig. 4.8 Generic NNI bearer release procedure.

At the UNI the procedure will be slightly different, because the BCA FEs will neither de-allocate bandwidth nor disconnect a switch path. Figure RG-1 of appendix B shows how release of connections takes place according to the above mentioned procedure. The corresponding activity in the user plane is depicted in figure 4.9.

4.2.5 Methods of Call Setup

4.2.5.1 Call Only

In section 4.2.1 the establishment of a call without any connections involved has been explained. The sequential setup of a call with one or more connections comes most close to this uncomplicated scenario.

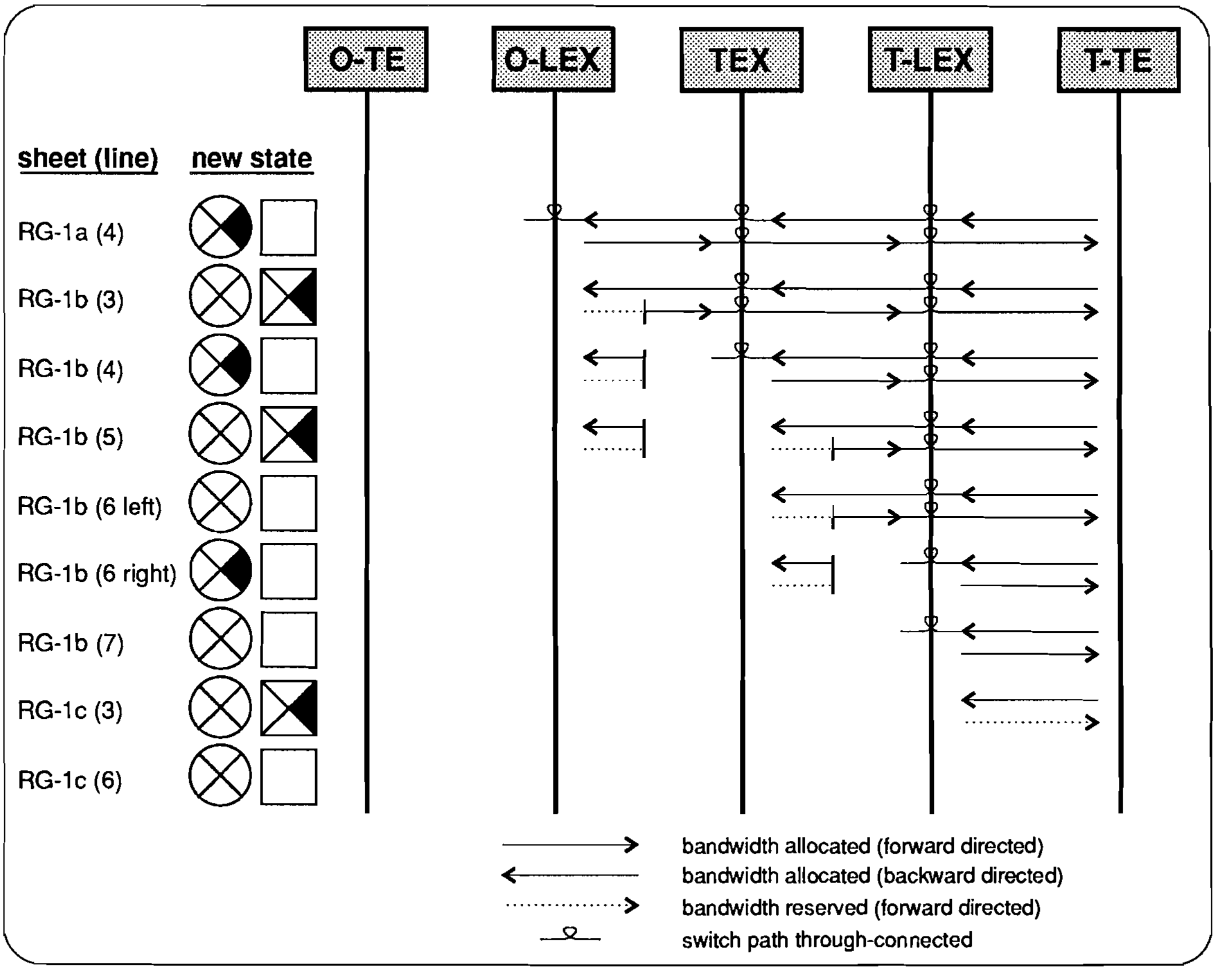

O-TE
O-LEX
TEX
T-LEX
T-TE
sheet (line)
new state
RG-1a (4)
RG-1b (3)
RG-1b (4)
RG-1b (5)
RG-1b (6 left)
RG-1b (6 right)
RG-1b (7)
RG-1c (3)
RG-1c (6)
bandwidth allocated (forward directed)
bandwidth allocated (backward directed)
bandwidth reserved (forward directed)
switch path through-connected

Fig. 4.9 Release of a call with one connection: how activity in the user plane should take place.

4.2.5.2 Sequential Setup

Sequential setup uses two steps: first the call will be set up, immediately followed by the establishment of the connection(s), as can be seen in figures B10-5 and B10-11 of appendix B. In the call control level end-to-end negotiation will take place. No bandwidth will be spoiled if, for example, the terminals are incompatible. The connections are set up after the called terminal confirmation is received by the calling terminal. The drawback of this way of call establishment is the so called "after dial delay." It will take some time before the call and the connections are set up.

If the called party is a mobile user, sequential setup has certain benefits: The CC information may travel a long, inefficient route through the network. But the BC information and the user data can be set up directly to the called party's current nodes. See also section 4.2.6.

4.2.5.3 Simultaneous Setup

Simultaneous establishment is a faster way to set up a call with some connections. The outgoing CC entities instruct the outgoing BC entities to setup up a bearer by means of a BEARER ENQ. req. ind. flow and wait for the BC entity's confirmation. Now both the CC FE and the BC FE may send a setup request information flow (CALL SETUP req. ind. and BEARER SETUP req. ind). In the protocol (stage 3!) these flows may be collected and sent within a single message.

The benefit of simultaneous setup, compared to sequential setup, is the shorter after dial delay. A drawback is the raised probability that bandwidth will be spoiled.

So the (human) user will prefer simultaneous setup above sequential setup because of speed, and the network operator will prefer sequential setup because of efficiency. Therefore it may be necessary to transform one setup method into an other method. Figure B10-14 of appendix B illustrates a call setup scenario using simultaneous setup at the O-UNI and sequential setup at the NNI and T-UNI.

4.2.5.4 Overlap sending

Overlap sending stands for sending the called user's address piece by piece and set up the call with the connections in portions. The information flow diagram in figure B10-10 of appendix B shows a scenario of a call being setup by overlap sending. Sometimes overlap sending is restrictively called digit-by-digit sending, but the address info send in each **ADDL. ADDR. req. ind.** information flow ("additional address info") is not limited to single digits.
The user provides some address information (the **ADDR. INFO** flows in figure B10-10a). The address information will be transferred through the network by means of **ADDL. ADDR. req. ind.** information flows up to the node whereto the call has been set up already. When a node has collected enough information to set up the next link a **C+B SETUP req. ind.** is sent from the incoming FE to the outgoing FE of that node. As soon as the terminating terminal has enough information to inform the user about the incoming call, a **REPORT (proceeding) req. ind.** will be sent back to the originating user indicating that the address is complete.

Overlap sending is on this moment an underexposed procedure within the standardization bodies. The currently accepted information flows diagram, which is depicted in figure B10-10, is very unstable. It is even not clear if an **ADDL. ADDR. req. ind.** flow should be sent to the TBCA and consequently if the resulting **REPORT (proceeding) req. ind.** flow should be initiated by the OTCC. This is the problem: the address is complete if the OTCC can uniquely identify a TCCA. Any additional information does belong to the sub-address, which is an supplementary service. But the information flow diagram in figure B10-10 describes a basic call scenario.

The major drawback of this way of setting up a call is the need of CC FEs in transit nodes, thus increasing the complexity and cost of transit exchanges. Also bandwidth will be allocated with the possibility that an illegal address will be dialled.
It is not possible to wait in the originating LEX until the entire address has been received, because there is (up to now) no way to check whether the last piece of addressing information has been received. However, following technique can be used: start a timer at the arrival of each **ADDL. ADDR. req. ind.** information flow. The address is assumed to be complete if no subsequent flow arrives before the timer expires. Of course a monstrous after dial delay will be introduced. After the timer expires – let's say 5 to 10 seconds – the entire call must be set up. The need of CC FEs in transit nodes can be averted by added functionality to the BC entities to let them deal with the **ADDL. ADDR. req. ind.** information flows.

Two important advantages of overlap sending are the very short after dial delay and the affiliation to release 1 B-ISDN, N-ISDN, and other networks. The evolution scenarios demand support of overlap sending by interworking units at least. The interworking unit may not know whether the address is complete. The network will still be able to support the call.

4.2.6 Roll-back mechanism

The roll-back mechanism, based on the "Commitment, Concurrency and Recovery" (CCR) protocol [18], [CCITT rec. X.850, X.851], is a technique which is specially useful for multiparty calls and for mobile called parties. The mechanism work as follows: First a request is sent the called party. The called party responds by accepting or rejecting optional parts c.q. the entire request. Finally the calling party decides what configuration will be used and informs the called party about it's decision. After these three steps the requested actions itself may be performed, requiring two additional steps in most cases: a requesting and a responding flow.

The information flow diagram in figure B10-15 of appendix B demonstrates a scenario with the roll-back mechanism. First the originating user sends an **ADD BEARER req. ind.** to the terminating CCA. The terminating CCA responds with an **ADD BEARER resp. conf.** information flow, whereupon the originating CCA instructs the BC level to set up the connection. So the third step of the roll-back mechanism is embedded in the bearer setup itself.
This shortened variant, consisting of two 2-step procedures, is called a look-ahead procedure. The third step of the roll-back mechanism is not explicitly described, but the information flow is implicit dispatched by the **BEARER SETUP req. ind.** flow in the BC level. This is an illegal technique in stage 2: every implicit information flow *is* an information flow and must be explicitly described. In stage 3 the flows may be gathered and transmitted in one single message. Sequential setup of a call also uses a look-ahead procedure. But the procedure is *only* used to set up the call. For the addition or modification of connections within an existing call (thus after the sequential setup has been completed) the roll-back mechanism is a serious option.

The difference between the roll-back mechanism and the bearer setup scenario described in section 4.2.2 is the moment that the **ADD BEARER resp. conf.** information flow is sent. In the normal bearer setup procedure this flow may be combined (stage 3!) with the **BEARER SETUP resp. conf.** flow and transmitted in one single message. In the look-ahead procedure the **ADD BEARER resp. conf.** must essentially be sent before the connection can be set up by the BC level. Hence an extra message must be transmitted through the network, introducing addition delay.
The benefits of the roll-back mechanism can be found in mobile users and multiparty calls. Just like sequential setup, the roll-back mechanism implies that connections can be set up directly to the current node of a mobile user. In case of multiparty calls a request is sent to all parties. The setup itself will take place after all parties have responded and the calling party has decided what configuration will be used.

4.3 Point-to-Multipoint with Branching Nodes

In a branching node user data is distributed without any modification, hence user data will have to be copied and in the control plane information flows will process differently. In point-to-multipoint scenarios one incoming information flow may cause several nearly identical outgoing flows. In the opposite direction some incoming information flows may be collected resulting in only one outgoing flow.
A branching node can be seen as a very simple bridge or as an extended connection. The latter point of view is most interesting, because this implies that a distributive connection comes under the basic call concept. One aspect has to be emphasized: Branching nodes involve uni-directional distributive connections only. So in one direction user data is distributed and in the opposite direction the bandwidth is zero.
Appendix B contains ten information flow diagrams showing scenarios with branching nodes based on the two level model (figures RGM-1 to RGM-10). Every information flow diagram describes three different locations of the branching node: within the O-LEX, TEX, and T-LEX.

A number of open issues have to be filled in, like where to split the CC and BC level, do distinctions exist between the CC and BC level, which new information flows are required, are all procedures reusable.

4.3.1 BC Level

The outgoing BC functional entities get an additional instance in the branching node. So a peer-to-peer association exists between BC entities of two adjacent nodes, corresponding to the link-by-link significance of associations in the BC level. This way of instantiating BC entities is also in alignment with the multi-connection BC instantiation policy for PP scenarios.
The consequence of this decision is that within a node a branch occurs in the BC level. Fortunately this comes very close to the user plane behaviour! Another consequence is that only intranode relationships will split, hence the procedures on the external relationships will need minimal modifications.

Considering the state-of-the-art switching hardware, the conclusion that branching nodes can be handled by the BC level can be easily drawn. Many switching architectures support user data duplication, e.g. by copying a portion of user data to several output buffers at once.

4.3.2 CC Level

The outgoing CC functional entities get no additional instance in the branching section. CC entities are far more complex than BC entities, so the number of instances should be minimized. Another reason to use a single instance is alignment with the three level model. Section 4.4 will discuss scenarios using the three level model.

Mark the term branching *section* (first line of previous paragraph) in stead of branching node, because branching in the BC level and the user plane may occur in a relaying node while no CC is present in that node. Figures RGM-ld and RGM-le of appendix B show different branching scenarios for the BC level, without differences in the CC level. In these figures the BC FEs are depicted but not used. The information flows are even identical. To avoid superfluous figures and to reduce redundancy in the information flow diagrams unused BC FEs will not be depicted. Compare figures RGM-1 and RGM-2; the figures depict the same scenario.

4.3.3 New Information Flows

Many information flows must be extended to support multiple destination addresses. Some additional information flows are required to add or remove a party or branch. In the CC level the following new information flows appear:

- **ADD PARTY req. ind.** is used to initiate a **CALL SETUP req. ind.** flow to the new party;
- **ADD PARTY resp. conf.** confirms the **ADD PARTY req. ind.** information flow;
- **PARTY SETUP req. ind.** is used to initiate the bearer establishment part of a sequential setup procedure;
- **PARTY SETUP resp. conf.** confirms the **PARTY SETUP req. ind.** information flow;
- **REMOVE PARTY req. ind.** initiates a **CALL RELEASE req. ind.** flow to the separating party;

- **REMOVE PARTY resp. conf.** confirms the **REMOVE PARTY req. ind.** information flow.

These information flows are transferred on the shared part of the network. After the branching node the normal **CALL SETUP** and **CALL RELEASE** information flows are used (see figures RGM-7 and RGM-8 of appendix B). Also **PARTY SETUP req. ind.** will result in **BEARER ENQ. req. ind.** or **BRANCH ENQ. req. ind.** flows (see figure RGM-9).
Following information flows are added on CC-BC relationships:

- **BRANCH ENQ. req. ind.** initiates a **BRANCH SETUP req. ind.** information flow. The latter flow will be transferred to a relaying branching node;
- **BRANCH CON. req. ind.** confirms the **BRANCH ENQ. req. ind.** flow;
- **BRANCH DEL. req. ind.** initiates a **BRANCH RELEASE req. ind.** information flow. The latter flow will be transferred to a relaying branching node.

These information flows are used to add or drop a branch in a relaying node without CC.
Following information flows, added in the BC level, are used for the same purpose (see also figures RGM-9 and RGM-10):

- **BRANCH SETUP req. ind.** initiates a **BEARER SETUP req. ind.** information flow in a relaying branching node;
- **BRANCH SETUP resp. conf.** confirms the **BRANCH SETUP req. ind.** flow;
- **BRANCH RELEASE req. ind.** initiates a **BEARER RELEASE req. ind.** information flow in a relaying branching node;
- **BRANCH RELEASE resp. conf.** confirms the **BRANCH RELEASE req. ind.** flow.

4.3.4 Procedures and Mechanisms

Simultaneous setup of a call is no serious option for multiparty scenarios, because called parties may reject different optional parts of a call. When sequential setup is applied, the responses can be collected before the final configuration is set up. Also look-ahead technique is required for a useful bearer addition procedure. Figure RGM-4, 5, 6, 9, and 10 of appendix B illustrate the use of look-ahead procedures.
Point-to-multipoint scenarios with branching nodes are not restricted to one single branching node per connection. A connection may have several branches.
A branch may be appended to uni-directional connections when a party is added in a two party call, but two dilemmas must be coped with:

- A bi-directional connection cannot branch, because data from a number of parties must be mixed in some way. A bridge is required or the connection must be divided into two uni-directional connections;
- None of the nodes the existing connection is routed along is able to support branching. In this case the connection must be re-routed.

Connection modification scenarios are not discussed in this report.

Conclusion: No new mechanisms are required for point-to-multipoint connections, but some existing establishment procedures cannot be used. Not all connection types may branch. Some hardware implementations may not support branching, so not every node can be a branching node.

4.4 Point-to-Multipoint with a Bridge

A bridge mixes user data and distributes the results. Figure 4.10 shows a block diagram of a bridge. The mixer needs to know the meaning of the user data, so the data stream must have a certain restricted format. Hence user data transparency is impossible in a bridge! The processing of user data by mixer may result in either one outgoing data stream for all users, or several different data streams.

The distributor does exactly the same as a branching node described in section 4.3: distribution of the user data by branching.

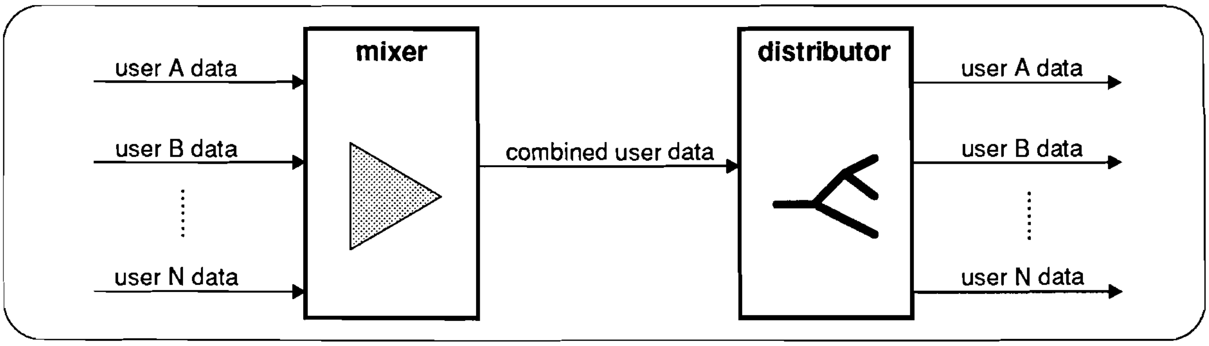


Fig. 4.10 Block diagram of a bridge.

In this report discussion about point-to-multipoint scenarios with bridges will only deal with non-distributed bridges. Also hand-over of a bridge will not be discussed. The former restriction implies that a bridge is always concentrated in one single node. The latter restriction implies that a bridge will stay in the same node during the entire call. Of course a call may have several MP connections, each of them having a bridge in a different node.

Most features of bridges can be made plausible without dealing with advanced scenarios. Figures RG3-1 to RG3-6 of appendix B show several information flow diagram of PM call scenarios having a bridge. The diagrams use the three level model with an intermediate special resource control level.

4.4.1 Activity in the RC Level

The special resource control level gets an instruction of the CC level by means of a **RESOURCE ENQ. req. ind.** information flow to set up a bridge, as can be seen in figure RG3-1 of appendix B. The OORC functional entity selects the location of the bridge. A bridge can be located in any node, depending on the node's capabilities and the allocation strategy.

The bridge sets up all involving bearers. So if the bridge is located in a relaying node, the bearer at the incoming NNI will be set up in reversed order. An exception forms the bearer at the O-UNI. This bearer will be set up partially before the **RESOURCE ENQ. req. ind.** flow, because the **B. SETUP req. ind.** information flow of this bearer causes the **RESOURCE ENQ. req. ind.** flow, and after the **RESOURCE CON. req. ind.** flow, because the **B. SETUP resp. conf.** flow confirms the **RESOURCE CON. req. ind.** information flow.

When the OOCC FE receives a **RESOURCE CON. req. ind.** information flow the bridge and all involving connections have been established successfully by the RC level.

Conclusions: The RC level is responsible for the allocation of the bridge and the establishment of the bridge and all involving connections. If the RC level is present no direct relationships exist between the CC and BC FEs. If a call has both 'conference' connections with bridges and P-P or P-MP connections a configuration as depicted in figure 4.11 may occur. So RC entities are used for connections with a bridge only.
For the RC level a number of new information flows are required to set up and release a bridge or branch.
A bridge is always controlled from the O-LEX, the node the owner of the call is connected to.

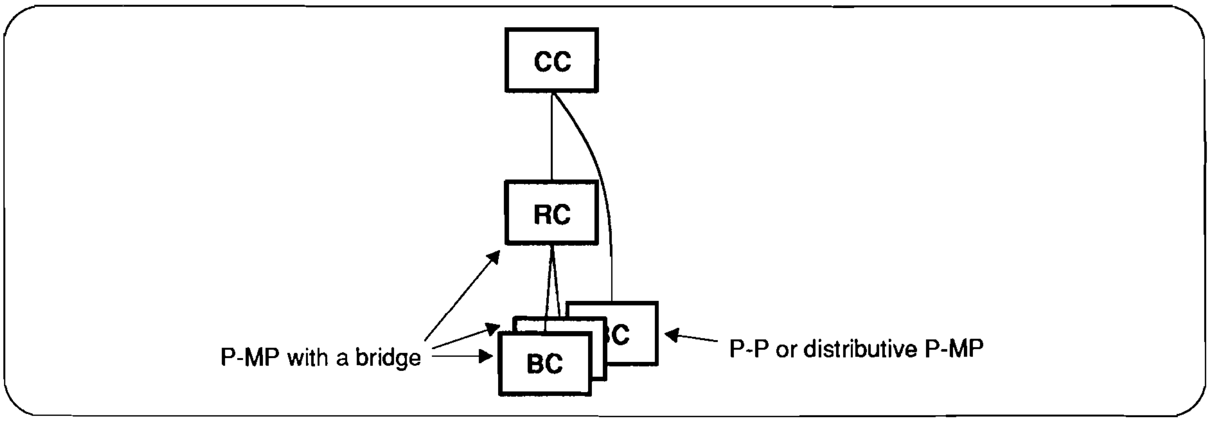


Fig. 4.11 A call configuration involving both bridges and simple P-P or uni-directional P-MP connections.

4.4.2 BC Level

In the optimal situation the BC level will not be aware of a bridge. Unfortunately through-connection in a node must be handling differently if a bridge is involved. An incoming bearer is not connected to an outgoing bearer, but both bearers are connected to the bridge. Thus both the incoming and outgoing BC FEs are responsible for through-connection.
Considering the AC model (and anticipating a little) thereby assuming that AC FEs are responsible for through-connection, a major profit of the model can be found for the BC level: The BC entities will only reserve, allocate, and release connections between two adjacent nodes, hence the BC level is not aware of any advanced configurations.

4.4.3 Procedures

To set up a call with point-to-multipoint connections with a bridge, the simultaneous setup procedure will be unusable for the same reasons as for point-to-multipoint scenarios with branching nodes.
Several new information flows are required – mainly in the RC level – to support bridges on top of the new information flows for branching nodes. A decision must be made whether connections are set up from the origination user to the other parties or all connections are set up from the bridge to all parties. The former procedure requires the connection between the originating user and the bridge to be set up first. I have chosen for the latter solution because all connections may be set up simultaneously by this procedure, thus reducing after dial delay.

4.4.4 Bridges

From the normal point of view a bridge consists of a mixer and a distributor. From a different standpoint a bridge can be seen a mixer only, whilst the distributor part is covered by the BC level and branching nodes. So the bridge has a reduced complexity and branching will be dealt with as normal distributive connections. This point of view excludes the existence of the MP–MP connection type, because the branching bearers have to be uni-directional.
The reduced bridge is not studied in this graduation task, but it may be an objective for future (short term) development.

4.5 Proposed Model with AC Entities

The basic functional model with AC entities is a likely candidate to replace both the two level model and the three level model with RC entities: It is not as complex as the three level model, and it satisfies the need of additional control facilities for multiparty calls.
Figure AC-1 and AC-2 of appendix B show two sequential call setup scenarios using the AC model. Figure AC-1 illustrates a two party setup with one connection. The internode procedures in this information flow diagram differ just a hairbreadth from the normal point-to-point sequential call setup procedures. However, within the nodes the AC entities take over the information flows between the CC and BC levels. Also the BC entities embody less functionality. The AC FEs are responsible for through-connection and disconnection of the switch path. So the BC FEs control a bearer between two adjacent nodes on a peer-to-peer basis. The AC entities may also take over some functions of the CC FEs.

Conclusions: The AC FEs are the nucleus of a node, containing shared functions and controlling through-connection of incoming and outgoing bearers. AC entities may be used as the interfacing point for intelligent network (IN) purposes.
An important purpose of BC and CC FEs is internode information interchange. The BC level also controls all connections and the CC level controls all telecommunication services aspects of the basic call.

Figure AC-2 of appendix B describes a three party call. This information flow diagram shows the advantage of relocating the through-connection functions to AC FEs: the BC entities do not have to be aware of the MP configuration. Only the AC FE in the relaying node acts differently, compared to figure AC-1. Now it controls both the switch path and the bridge. The extra information items required to setup and control the bridge are transferred by BC level information flows. BC entities may pass these additional items transparently.

A drawback of the AC model is the extra internal relationships compared to the two level functional model. In stage 3 only the specification of external relationships is required to define a protocol. Internal relationships are merely suited as a guideline for network equipment manufacturers: ignoring the internal node specification may cause a bunch of misery when dealing with standardized expansions in the future.

4.6 Individual Information Flow Descriptions

In this section a number of individual information flows are described in detail. The information flows are based on the two level functional model. Table 4.1 shows which information flows are described. Also specified in the table are the relationships the information flows appear within, as well as the direction. Note that overlap sending and relaying CC will not be considered.

Table 4.1 Overview of the information flows described in this section. After each information flow the direction and the relationships in which the flow appears are given.

Flows in CC-BC relationships

BEARER ALLOCATION req. ind.	fwd	r7,r8
BEARER CONNECTED req. ind.	bk	r1,r3,r7
BEARER DELETE req. ind.	fwd	r1,r2,r3,r6,r7,r8
BEARER ENQUIRY req. ind.	fwd	r1,r3,r7
BEARER ENQUIRY resp. conf.	bk	r1,r3,r7
BEARER ESTABLISH req. ind.	fwd	r2,r6,r8
BEARER ESTABLISH resp. conf.	bk	r2,r6,r8

Flows in internode relationships

ADD PARTY req. ind.	fwd	r11,r10
ADD PARTY resp. conf.	bk	r11,r10
BEARER ALLOCATION req. ind.	fwd	r26
BEARER RELEASE req. ind.	fwd	r21,r22,r25,r26
BEARER RELEASE resp. conf.	bk	r21,r22,r25,r26
BEARER SETUP req. ind.	fwd	r21,r22,r25,r26
BEARER SETUP resp. conf.	bk	r21,r22,r25,r26
CALL ALLOCATION req. ind.	fwd	r17
CALL RELEASE req. ind.	fwd	r11,r10,r17
CALL RELEASE resp. conf.	bk	r11,r10,r17
CALL SETUP req. ind.	fwd	r11,r10,r17
CALL SETUP resp. conf.	bk	r11,r10,r17
C+B SETUP resp. conf.	bk	r11,r10
RELEASE CONFIRM req. ind.	fwd	r11,r10,r17
RELEASE CONFIRM resp. conf.	bk	r11,r10,r17
REMOVE PARTY req. ind.	fwd	r11,r10
REMOVE PARTY resp. conf.	bk	r11,r10
REPORT (alerting) req. ind.	bk	r11,r10,r17
REPORT (proceeding) req. ind.	bk	r11

Remaining information flows

BEARER REPORT req. ind.	both	all relationships
CALL REPORT req. ind.	both	r11

Flows in intranode relationships

ADD PARTY req. ind.	fwd	r12,r16
ADD PARTY resp. conf.	bk	r12,r16
BEARER RELEASE req. ind.	fwd	r12,r23,r16
BEARER RELEASE resp. conf.	bk	r23
BEARER SETUP req. ind.	fwd	r12,r23,r16
BEARER SETUP resp. conf.	bk	r12,r23,r16
CALL RELEASE req. ind.	fwd	r12,r16
CALL SETUP req. ind.	fwd	r12,r16
CALL SETUP resp. conf.	bk	r12,r16
C+B RELEASE req. ind.	fwd	r12,r16
C+B SETUP req. ind.	fwd	r12,r16
C+B SETUP resp. conf.	bk	r12,r16
REMOVE PARTY req. ind.	fwd	r12,r16
REPORT (alerting) req. ind.	bk	r12,r16

Flows in User-CC relationships

ADD PARTY conf.	bk	O-user
ADD PARTY req.	fwd	O-user
ALLOCATE conf.	bk	O-user
ALLOCATE ind.	fwd	T-user
ALLOCATE req.	fwd	O-user
ALLOCATE resp.	bk	T-user
DEALLOCATE ind.	fwd	T-user
DEALLOCATE req.	fwd	O-user
RELEASE ind.	fwd	T-user
RELEASE req.	fwd	O-user
REMOVE PARTY req.	fwd	O-user
REPORT ind.	bk	O-user
SETUP conf.	bk	O-user
SETUP ind.	fwd	T-user
SETUP req.	fwd	O-user
SETUP resp.	bk	T-user

CALL SETUP req. ind. is used to request establishment of a call. This is a confirmed information flow and **CALL SETUP resp. conf.** is used to confirm that the call has been established. The request for establishment of a call can be originated by the user. This information flow appears within the r10, r11, r12, r16, r17 relationships. The following items of information are or may be conveyed with the **CALL SETUP req. ind.** and **CALL SETUP resp. conf.** information flows:

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*[1]
protocol info	ID10	mandatory	mandatory	
	bridge capabilities	optional		note 1
bearer info	bearer capabilities	mandatory		
	nature of transmission	mandatory		
	ATM attributes (in ATM)	mandatory	mandatory	
routing info	called number	mandatory		(n times)[1]
	transit network selection	optional		
originating info	called line ID	optional		

	connected line ID request	optional		
terminating info	connected line ID		optional	
	connected line status		optional	
access info	AAL attributes (in ATM)	mandatory	mandatory	
	low layer compatibility	optional	optional	
	high layer compatibility	optional	optional	

NOTE 1: bridge functionality depends on the number of called users, the nature of transmission, and other user request.

CALL SETUP resp. conf. is used to confirm that the call has been established. This information flow appears within the r10, r11, r12, r16, r17 relationships. See also **CALL SETUP req. ind.**

CALL ALLOCATION req. ind. is used to acknowledge that a previously sent **CALL SETUP resp. conf.** has been received and accepted. This is an unconfirmed information flow within the r17 relationship and is sent from the OTCC functional entity to the TCCA functional entity.

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID10	mandatory		

CALL REPORT req. ind. is an information flow that is used to report status and/or other types of information across the network. The type of information may be indicated, e.g. alerting, suspending, hold, resume, proceeding, etc. This is an unconfirmed information flow that appears within the r10, r11, r12, r16, r17 relationships. The following items of information are or may be conveyed with the **CALL REPORT req. ind.** information flow:

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID10	mandatory		
	report type	mandatory		
	connection request	optional		
general	called line category	mandatory		
	called line status	mandatory		
	report type specific info	optional		

REPORT (proceeding) req. ind. is an unconfirmed information flow that is used to indicate that sufficient address information (or digits) have been received to process a call attempt. This is an information flow within the r11 relationship. See **CALL REPORT req. ind.** for complete description.

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID10	mandatory		
	address complete	optional		

REPORT (alerting) req. ind. is an unconfirmed information flow that is used to indicate that TCCA functional entity has received and accepted an establishment request and now waits for the user to respond. This is an unconfirmed information flow that appears within the r10, r11, r12, r16, r17 relationships. See **CALL REPORT req. ind.** for complete description.

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID10	mandatory		

BEARER SETUP req. ind. is used to request establishment of a connection. This is a confirmed information flow and **BEARER SETUP resp. conf.** is used to confirm that the connection has been established. The request for establishment of a connection can be originated by either the user or the network. This information flow appears within the r12, r16, r21, r22, r23, r25, r26 relationships. The following items of information are or may be conveyed with the **BEARER SETUP req. ind.** and **BEARER SETUP resp. conf.** information flows:

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID2	mandatory	mandatory	
	ID10	mandatory	mandatory	
	connection request	optional	optional	
bearer info	ID1	mandatory[1]	optional[1]	
	bearer capabilities			
	nature of transmission			
	time slot sequence info	optional		
	ATM attributes (in ATM)	mandatory	mandatory	
routing info	called number	mandatory		(n times)
	transit network selection	optional		
terminating info	connected line status		mandatory	
access info	AAL attributes (in ATM)	mandatory	mandatory	
	low layer compatibility	optional	optional	
	high layer compatibility	optional	optional	

NOTE 1: not present in the r1 relationship.

BEARER SETUP resp. conf. is used to confirm that the connection has been established. This information flow appears within the r12, r16, r21, r22, r23, r25, r26 relationships. See also **BEARER SETUP req. ind.**

BEARER ENQUIRY req. ind. is used by a CC functional entity to request a BC functional entity establishment of a connection. This is a confirmed information flow if establishment of the call and the connection will be simultaneously (in this case **BEARER ENQUIRY resp. conf.** is used to confirm that the BC functional entity might be able to establish a connection), else this is an unconfirmed information flow. This information flow appears within the r1, r3, r7 relationships. (See also **BEARER CONNECTED req. ind.**)

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	response expected	optional		
bearer info	ID1	mandatory[1]	optional[1]	

NOTE 1: not present in the r1 relationship.

BEARER ENQUIRY resp. conf. is used to confirm that the BC functional entity has received and accepted the enquiry request. This information flow appears within the r1, r3, r7 relationships. See also **BEARER ENQUIRY req. ind.**

BEARER ESTABLISH req. ind. is used by a BC functional entity indicate a CC functional entity the establishment of a connection. This is a confirmed information flow and **BEARER ESTABLISH resp. conf.** is used to confirm that the connection has been established. This information flow appears within the r2, r6, r8 relationships.

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
bearer info	ID1	mandatory	optional	

BEARER ESTABLISH resp. conf. is used to confirm that the connection has been established. This information flow appears within the r2, r6, r8 relationships. See also **BEARER ESTABLISH req. ind.**

BEARER CONNECTED req. ind. is used by a BC functional entity to indicate a CC functional entity the establishment of a connection. This is an unconfirmed information flow that appears within the r1, r3, r7 relationships.

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
bearer info	ID1	optional		

BEARER ALLOCATION req. ind. is used to acknowledge that a previously sent **BEARER SETUP resp. conf.** has been received and accepted. This is an unconfirmed information flow within the r7, r8, r26 relationships.

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID2	mandatory		
	ID10	optional		

BEARER REPORT req. ind. is an information flow that is used to report status and/or other types of information across the network. The type of information may be indicated, e.g. alerting, suspending, hold, resume, etc. This is an unconfirmed information flow that appears within all relationships. The following items of information are or may be conveyed with the **BEARER REPORT req. ind.** information flow:

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID2	mandatory		
	ID1	optional		
	ID10	optional		
	report type	mandatory		

	connection request	optional
general	called line category	mandatory
	called line status	mandatory
	report type specific info	optional

C+B SETUP req. ind. is used to request simultaneously establishment of a call with a connection. This is a confirmed information flow and **C+B SETUP resp. conf.** is used to confirm that the call and the connection have been established. The request for establishment of a call can be originated by the user. This information flow appears within the r12, r16 relationships. (See also **CALL SETUP req. ind.** and **BEARER SETUP req. ind.**)

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*[1]
protocol info	ID10	mandatory	mandatory	
	connection request	optional	optional	
	bridge capabilities	optional		note 1
bearer info	ID1	mandatory	optional	
	bearer capabilities	mandatory		
	nature of transmission	mandatory		
	time slot sequence info	optional		
	ATM attributes (in ATM)	mandatory	mandatory	
routing info	called number	mandatory		(n times)[1]
	transit network selection	optional		
originating info	called line ID	optional		
	connected line ID request	optional		
terminating info	connected line ID		optional	
	connected line status		mandatory	
access info	AAL attributes (in ATM)	mandatory	mandatory	
	low layer compatibility	optional	optional	
	high layer compatibility	optional	optional	

NOTE 1: bridge functionality depends on the number of called users, the nature of transmission, and other user request.

C+B SETUP resp. conf. is used to confirm that a call with a connection have been established. This information flow appears within the r10, r11, r12, r16 relationships. See also **C+B SETUP req. ind.**

CALL RELEASE req. ind. is used to request release of a call. Within the r10, r11, r17 relationships this is a confirmed information flow and **CALL RELEASE resp. conf.** is used to confirm that the call release request has been received and accepted. Within the r12, r16 relationship this is an unconfirmed information flow.

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID10	mandatory	mandatory	

CALL RELEASE resp. conf. is used to confirm that a call release request has been received and accepted. This information flow appears within the r10, r11, r17 relationships. See also **CALL RELEASE req. ind.**

BEARER RELEASE req. ind. is used to request release of a connection. Within the r21, r22, r23, r25, r26 relationships this is a confirmed information flow and **BEARER RELEASE resp. conf.** is used to confirm that the connection release request has been received and accepted. Within the r12, r16 relationships this is an unconfirmed information flow.

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID2	mandatory	mandatory	
	ID10	mandatory	mandatory	

BEARER RELEASE resp. conf. is used to confirm that the connection release request has been received and accepted. This information flow appears within the r21, r22, (r23,) r25, r26 relationships. See also **BEARER RELEASE req. ind.**

BEARER DELETE req. ind. is used by a CC functional entity to request a BC functional entity release of a connection and is used by a BC entity to indicate a CC entity that a connection has been released. This is an unconfirmed information flows that appears within the r1, r2, r3, r6, r7, r8 relationships.

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	? ID10	mandatory	mandatory	

RELEASE CONFIRM req. ind. is used to request the confirmation of a connection release. This is an confirmed information flow and **RELEASE CONFIRM resp. conf** is used to confirm the requested release. The information flow appears within the r10, r11, r17 relationships.

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID10	mandatory	mandatory	

RELEASE CONFIRM resp. conf. is used to confirm the requested release. The information flow appears within the r10, r11, r17 relationships. See also **RELEASE CONFIRM req. ind.**

C+B RELEASE req. ind. is used to request simultaneously release of a connection and a call. This is an unconfirmed information flow that appears within the r12, r16 relationships.

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID10	mandatory	mandatory	

ADD PARTY req. ind. is used to add a party to an existing call. This is an confirmed information flow and **ADD PARTY resp. conf.** is used to confirm that the call has been extended to the new party. This information flow appears in the r10, r11, r12, r16 relationships, but only if the existing call has been set up via that same route. Else the **ADD PARTY req. ind.** will be changed by the CC functional entity to a **CALL SETUP req. ind.** or a **C+B SETUP req. ind.**

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*[1]
protocol info	ID10	mandatory	mandatory	
	bridge capabilities	optional		note 1
routing info	called number	mandatory		(n times)[1]
	transit network selection	optional		
originating info	called line ID	optional		
	connected line ID request	optional		
terminating info	connected line ID		optional	
	connected line status		optional	
access info	AAL attributes (in ATM)	mandatory	mandatory	
	low layer compatibility	optional	optional	
	high layer compatibility	optional	optional	

NOTE 1: bridge functionality depends on the number of called users, the nature of transmission, and other user request.

ADD PARTY resp. conf. is used to confirm that the call has been extended to the new party. This information flow appears in the r10, r11, r12, r16 relationships. See also **ADD PARTY req. ind.**

REMOVE PARTY req. ind. is used to remove a party from an existing call. This is an confirmed information flow and **REMOVE PARTY resp. conf.** is used to confirm that the call has been shrunk. This information flow appears in the r10, r11, r12, r16 relationships, but only if the existing call has been set up via that same route. Else the **REMOVE PARTY req. ind.** will be changed by the CC functional entity to a **CALL RELEASE req. ind.** or a **C+B RELEASE req. ind.**

Use	*Item*	*req. ind.*	*resp. conf.*	*multi-destination*
protocol info	ID10	mandatory	mandatory	

REMOVE PARTY resp. conf. is used to confirm that the call has been shrunk. This information flow appears in the r10, r11 relationships. See also **REMOVE PARTY req. ind.**

4.7 Conclusions

The information flow diagrams describing point-to-point scenarios are quite stable, but other information flow diagrams describing MP scenarios are very unstable.

The release of bandwidth must be revised in order to create an unambiguous user plane state in the setup and release procedures.

The information flows described in stage 2 may not be mapped on stage 3 messages in advance. So implicit information flows must be explicitly described, although the information flows may be collected and sent as one single message in stage 3.
Several different call setup procedures exist. To support all procedures CC FEs need to be rather complex. Simultaneous setup and overlap sending are even unusable for MP configurations.
No new mechanisms are required for point-to-multipoint connections, but some existing establishment procedures cannot be used. Bi-directional connections cannot branch, hence a bridge is mandatory in this case. Some hardware implementations may not support branching, so not every node can be a branching node.

The RC level is responsible for the allocation of the bridge and the establishment of a bridge and all involving connections. If the RC level is present no direct relationships exist between the CC and BC level. In the three level model the BC FEs must be aware of MP configurations in order to control switch path through-connection and disconnection accurately.
The distributing section of a bridge may be seen as a separated branching function. This point of view excludes the existence of the MP–MP connection type, because the branching bearers must be uni-directional.

The AC FEs are the core of a node, containing shared functions and controlling through-connection of incoming and outgoing bearers. AC entities may be used as the interfacing point for IN purposes.
Within the AC model an important purpose of BC and CC FEs is internode information interchange. The BC level also controls all connections and the CC level controls all telecommunication services aspects of the basic call.

Most information flows for successful two-party basic call scenarios have been studied in detail. A number of mandatory and optional information items have been identified.

5 SDL Diagram

5.1 Introduction

An information flow diagram shows the actions and information flows which have to be completed for one scenario. After comparing a lot of information flow diagrams, a overview arises of the total collection of functions and information flows each FE has to deal with.
Of course it is impossible to cover all possible scenarios. Fortunately nobody ever wants to describe all possible scenarios with information flow diagrams: The SDL diagrams are suited for these descriptions. SDL is a formal language for the specification and description of systems. It has been standardized by CCITT [10].
An SDL diagram describes a system embedded into its environment. The environment transfers information flows to the system and vice versa. An information flow will also initiate actions and new information flows in the system. The SDL diagram specifies which actions a system, or parts of the system, will apply.

This report will not go into the syntaxis of SDL. A plain introduction to SDL can be found in a book of F. Belina et al [19].

5.2 SDL Diagram of a Two Level B-ISDN Network

Appendix C contains a SDL diagram of a B-ISDN network, based on the two level model. The diagram has been derived from a lot of information flow diagrams. First the system is subdivided into nodes, called blocks. Then the nodes are divided into functional entities, called processes. A process may include some services. On the lowest level specify flow diagrams the exact behaviour of each piece of the system.

The SDL description covers point-to-point scenarios, multiple responding terminals, point-to-multipoint scenarios, and MP calls with some sort of bridge in the O-LEX.
In this investigational diagram the control of the bridge is assumed to be handled by the CC level. For this purpose the CC FEs in the O-LEX are cut into two portions: a typical CC entity and a Mix Control Unit (MCU).

Two side-notes must be added to the SDL diagram:

- The diagram forms an overview of the network. The syntax may be incorrect in some point (simplified notations), and semantically are many parts missing (like formal description of signals).
- The diagram is merely based on obsolete information flow diagrams, so a discrepancy exists with the information flow diagrams in appendix B.

The goal of the SDL diagram is investigation of the complexity of entities. Taking into account the above mentioned site-notes, this is a workable proposition.

5.3 Conclusions

5.3.1 General

The functional entities in the BC level are far less complex than FEs in the CC level. The CC FEs at the UNI require more functions than CC FEs at the NNI, because the UNI is less reliable.

Branching nodes are controlled by the incoming side of a node. Observe the IRBC and ORBC processes. The incoming BC process selects and instantiates the outgoing BC process. If the IRBC decides that user data has to be transported in several directions, several ORBC processes are started (one process per outgoing link). So the outgoing BC entities don't require branching functions. The incoming BC entities will control the user data copy features of the node.

The control of a bridge adds substantial complexity to the CC entities.

5.3.2 Call Setup Procedures

The major complexity in a CC process is caused by the existence of several setup procedures. The setup procedures differ quite a lot, so the process forks almost immediately after instantiation. The branches meet not before the setup phase has completed.
A reduction of complexity can be achieved by converting one setup procedure to another. This may be realized by IOCC. The IOCC entity will grow more complex, but all other CC entities in the network will get simpler, including the OCCA because this entity needs to support one procedure less.
Figure B14 of appendix B illustrates conversion from simultaneous setup to sequential setup. Another reason to support procedure conversion comes from the B-ISDN evolution scenario: release 1 equipment incapable to handle sequential setup. So a release 1 terminal will always try to establish a call and its connections simultaneously. The network may use sequential setup regardless of the procedure used at the UNI.

Note: In the AC model procedure conversion may be managed by AC entities, so the CC FEs may concentrate on controlling the call.

5.3.3 Control of Branching by Relaying BC

When an IRBC process receives a **BEARER_SETUP_req_ind** signal, the destination address must be inspected to determine which ORBC process must be instantiated (routing).

When the signal contains several destination addresses all addresses must be examined. For each destination an ORBC instance may be created, receiving a signal which contain only one address.

The differences between point-to-point and branching basic call scenarios seem to be diminutive.

Often information (user data and control information) may be partly transferred via the same route through the network. In this case only one signal with several addresses satisfies. Of course fewer instances are required to control fewer links.

5.3.4 Control of a Bridge

In the SDL description the distributing section of the bridge is integrated as branching nodes. The O-LEX contains just the mixing section, called Mix Control Unit (MCU). The MCU is poorly specified in the diagram, so conclusions must be tread warily.

Controlling a bridge add a lot of complexity to the CC level. So the appellation 'Call Control' is slightly out of track. The three level functional model and the AC model preserve this inconsistency.

6 Q.73 Demonstrator

6.1 Goal

The Q.73 demonstrator must meet the following basic requirement:

- The demonstrator must show how a control system with separated call control and connection control operates;
- The demonstrator must show the benefits of the separation between call control and connection control;
- The demonstrator must be easy to work with;
- Several scenarios must be demonstrated, e.g. call setup, connection setup within an existing call, simultaneous setup of a call with one or more connections;
- The demonstrator must be easily extendable for new scenarios.

The goal of the demonstrator is tripartite:

- Demonstration of the exchange of signalling information and actions in the functional model for B-ISDN signalling release 2;
- Helping hand for the explanation and understanding of the functional modelling for B-ISDN signalling release 2;
- Simulation of (sets of) scenarios for test purposes.

In the next sections the requirements of the demonstrator will be elaborated.

6.2 User Requirements

6.2.1 Language Choice

The language in the demonstrator programme will be English, because the demonstrator should be suitable for international purposes.

6.2.2 Environment

The demonstrator will be developed on a IBM compatible PC. On this computer system two environments are commonly available: DOS and Windows. The demonstrator will be developed to execute in the less complex DOS environment to reduce time needed for programming and to lower the PC system requirements.

6.2.3 User's Profile

There are two kinds of users: "presenters" and "programmers." The first group only want to *use* the demonstrator in the most easy and reliable way. The programmer installs, configures and optimizes the demonstrator and includes new scenarios. It will be clear that the demonstrator must be flexible enough for the second group of users and simple enough for the first group.

6.2.4 Interaction Device

Four options exist for user interaction: keyboard only, mouse only, both keyboard and mouse, mouse and/or keyboard where most efficient. The third option would be the most convenient option for the user point of view. For the developer of the program the last option is most simple and clearly structured. The first option, keyboard only, is an old fashioned solution which will not be considered from the author's diligence point of view. The mouse only option will be waste of time and maybe complex to develop. So the mouse and/or the keyboard where most efficient will be used.

Because a mouse has at least two buttons, two special functions may be defined: the left button for pointing objects on the screen and the right button for another function, like context sensitive help text.

6.2.5 Objects to be Demonstrated

The following must be displayed by the demonstrator:

- A single scenario, optionally followed by another scenario without clearing the screen, et cetera;
- A set of scenarios – a scenario group – to be displayed subsequent, like "establish call," "add one connection," "simultaneous release of call and connection;"
- Zoom into a functional entity and/or an information flow;
- Theory about the functional model and about the separation between CC and BC;
- Endless demo.

6.2.6 General Screen Lay Out

The most structured way to display data on a screen can be achieved by using windows, menus, and buttons. Figure 6.1 shows a possible screen lay out for the demonstration of a scenario.

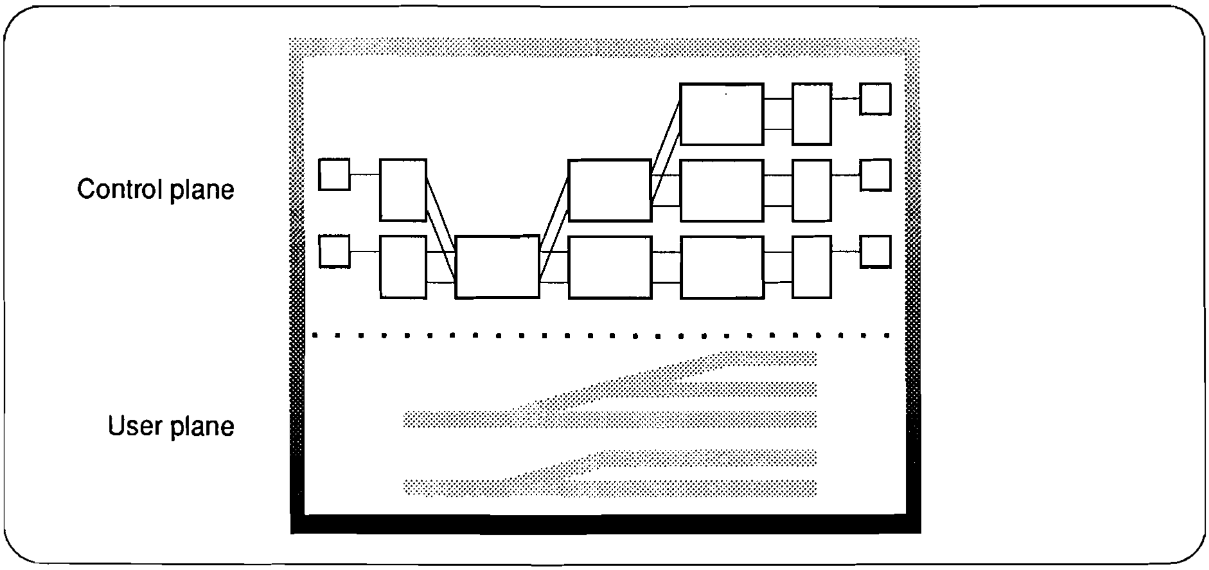


Fig. 6.1 Possible screen lay out for demonstration of a scenario.

6.2.7 How to Demonstrate a Scenario

On the screen the functional model and the user plane must be displayed. The model must be simplified as much as possible. The user plane can be visualized by tubes for example. An information flow may be displayed by an illuminating arrow having a few labels. An action may be displayed by actions in the user plane or by action codes and a short description. The next section will go deeper into this subject.

6.2.8 Interaction with the "Presenter"

The presenter type of users need following functions:

- Selection of a scenario;
- Addition of another scenario to the current displayed status;
- Simple creation of scenario groups;
- Selection of a scenario group;
- Zoom into a PE or FE.

6.2.9 Interaction with the "Programmer"

The programmer type of users will need the following additional functions:

- Installation of the demonstrator;
- (Re-)configure the demonstrator;
- Remove, include, or change the scenarios;
- Use other (?) test features.

6.3 Technical Requirements

6.3.1 System Requirements

The PC will have to meet the following needs:

- IBM compatible PC/AT (80286 cpu at least);
- MSDOS 3.30 or newer (or compatibles: PCDOS 3.30 / DRDOS 5.0);
- VGA compatible colour display, supporting graphics resolution 640x480x16;
- 640kB basic memory.

6.3.2 Programming Language

Three programming languages can be taken in consideration both for DOS and for Windows: Pascal, C, and C++. Because the development of the demonstrator is not a programming exercise, the language that will be used is the one most familiar to the author: Pascal. The demonstrator will be developed in Borland Pascal 7.0 for real-mode DOS.

6.3.3 Type of Scenarios

The demonstrator must demonstrate a number of scenarios. This may be translated as: at least the demonstrator must be able to simulate a wide range of basic call scenarios. If the scenarios will be included as external ASCII source files, the inclusion of new scenarios will be most simple.

6.3.4 Configuration

The demonstrator must be easy to configure from the programmer user point of view. There are two possibilities: the user edits an ASCII file containing settings, or the user configures the demonstrator interactively. The first option brings on the need of consultation of the user's manual when configuring the demonstrator. The second option is far more user friendly.

6.3.5 Lay Out of the Program

Figure 6.2 shows a simple flow chart of the demonstrator. All basic features of the program are depicted in this chart.

6.3.6 Setup Parameters

The question is how flexible the demonstrator must be. Things like which directory contains the source files, how fast must be simulated must have a default value. Other things like which colours or typeface to be used on the screen don't need to be adjustable. So a minimum set of mandatory parameters and some optional parameters can be identified.

Mandatory are: directory containing scenarios, wild card to select scenarios, directory containing functional models, wild card/file name to select functional model, what to demonstrate in the endless demo, how to interpret actions, how to interpret the user plane, simulation speed, sound on/off.

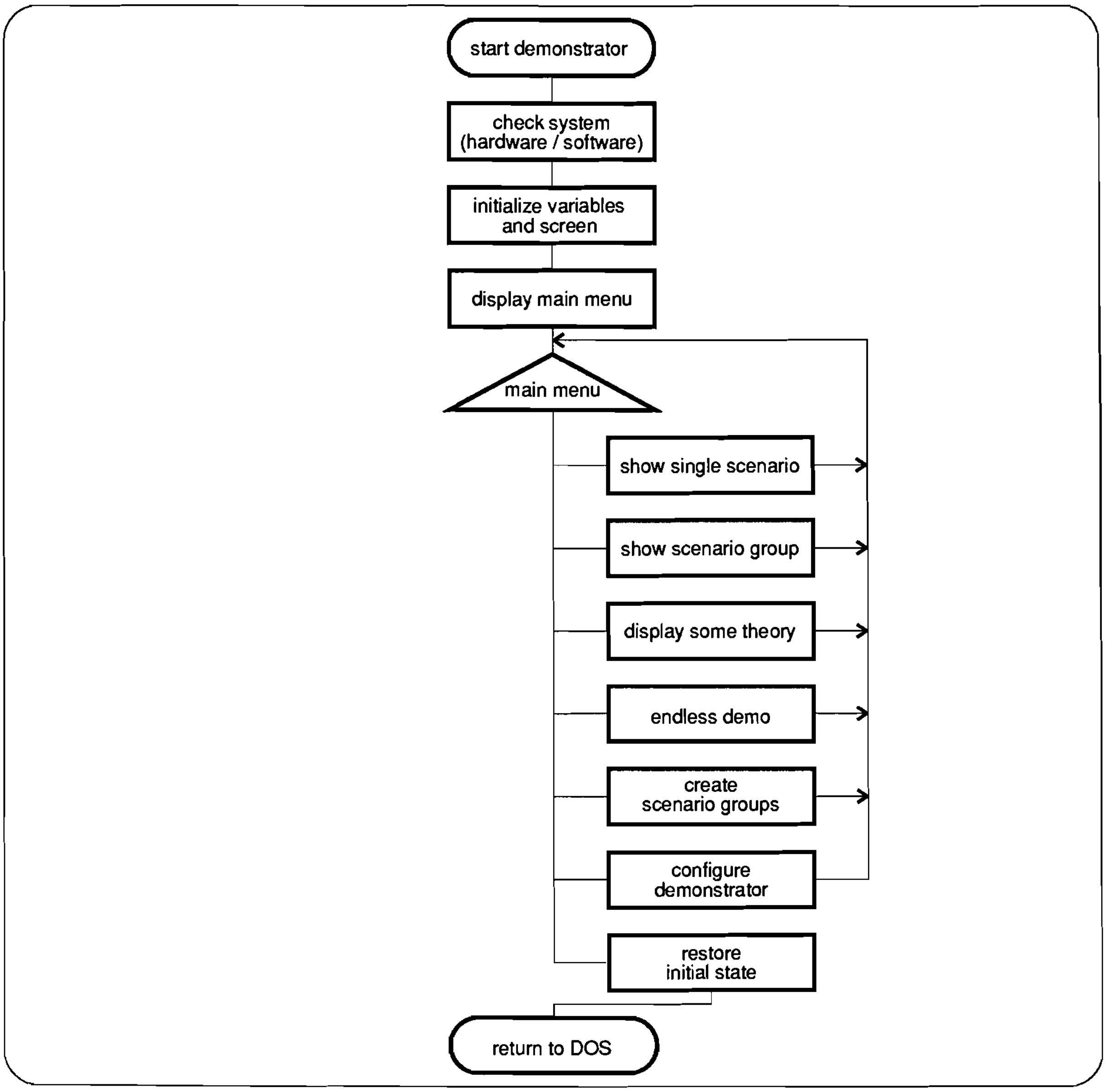


Fig. 6.2 Simple flow chart of the Q.73 demonstrator.

Some optional parameters: how to display the functional model, show actions yes/no, show users plane yes/no, interpret synchronization symbols yes/no.

6.4 Basic Procedures

A set of procedures and functions can be identified that will have to be developed to support all basic functions within the program. Of course the set is strongly dependent on the final program design. The list below shows some required procedure and functions. Of course this is *not* a complete list.

```
function init_chkDOS: boolean;
```

Checks if the correct DOS version is active. Determines the directory which contains the program itself. Determines the current directory and drive. Returns 'FALSE' on any error.

```
function init_chkVGA: boolean;
```

Checks if a VGA compatible display is available. Installs the VGA graphics driver and the screen fonts. Returns 'FALSE' on any error.

```
procedure init_chkMuis;
```

Checks if a mouse is available and resets the mouse driver and – if necessary – the mouse itself. The mouse cursor must be hidden and redefined. The mouse type will be stored in a variable.

```
procedure init_ReeksMemorize;
```

Saves the current VGA card palette state and DAC state. Necessary because the demonstrator will make use of colour fading which destroys the VGA colour state.

```
procedure init_ReeksRecall;
```

Restores the initial VGA card palette state and DAC state before returning to DOS.

```
procedure init_Heap;
```

Reserves space on the heap for screen grabbing, emulated text cursor, text files, et cetera.

```
procedure init_Layout;
```

Builds an empty screen and disable the mouse and text cursors.

```
procedure SetDefaultDac;
```

Selects the program's standard palette.

```
procedure ColorFade (speed: byte);
```

Fades the current screen colours to a new set of colours with the given speed. If the speed is 0 the colours will be changed immediately.

```
procedure MuisCrsr (action: byte);
```

Controls the graphics mouse cursor. Possible actions are: 'cursor on,' 'cursor off,' or 'restore previous state.'

```
procedure TextCrsr (action: byte);
```

Controls the emulated text cursor on the graphics screen. The procedure assumes the current line colour to be the foreground text colour. Possible actions are: 'cursor on,' 'cursor off,' or 'restore previous state.'

```
procedure Button (x,y,dx,dy: integer; shortcut: shortint; name: string);
```

Places a button on the screen and centres the name in it. A shortcut key may be specified. With 'shortcut' a the given character may be imaged in red and underlined. If −1 is specified the entire name will be underlined.

```
procedure Window (x,y,dx,dy: integer; name: string);
```

Places an empty window on the screen with a title and a 'Close' button.

```
procedure BoxMemorize (x,y,dx,dy: integer);
```

Saves the screen data of a specified area.

```
procedure BoxRecall;
```

Restores the screen data of a previously saved area.

```
function InPressed: boolean;
```

Tests for keyboard key-press or mouse button-press. If 'TRUE' the character of the coordinates will be evaluated and stored in variables.

```
function InReleased: boolean;
```

Tests if all mouse buttons are released. If 'TRUE' for the first time, the release coordinates will be evaluated and stored in variables.

```
function InKey: char;
```

Waits for a keyboard key to be pressed.

```
function InHot: byte;
```

Waits for a mouse click within a sensitive (hot) area on the screen. The area number is returned.

`ReadFile` and other procedures extracted from FLOWS. These procedures are required to read a information flow diagram file.

```
procedure DisplayModel;
```

Displays the memorized functional model.

6.5 Scenario Description

It's preferable that scenarios are described by text files having the same format as used by the Q.73 information flow diagrams imaging utility FLOWS [15] (see also chapter 4). Most simple would be that the FLW files can be interpret immediately, though some additional information is required:

- The lay out of the complete basic functional model;
- Information about addition and removal of entities caused by the scenario;
- State of the user plane at the start of the scenario;
- Changes to the user plane.

To avoid a lot of extra work, it is preferable to place the description of the basic functional model in a separate text file. So the scenario description requires just a reference to a model. Information about changes to entities and to the user plane have to be inserted in the file. This can be done by introducing a new item in the FLW files [15]. The initial state of the user plane will not be necessary when the demonstrator handles sets of scenarios with two restrictions:

- The first scenario in the 'group' initiates the call;
- For each following scenario in the 'group' the initial state of the user plane must *exactly* match the ending state of the predecessor.

6.6 Conclusions

All requirements concerning flexibility can be met by using text files for the description of scenarios. Regarding the developments in CCITT, it is wise to describe the basic functional model in a text file too. Plain FLW files don't describe all the information required for simulation. The user and technical requirements worked out in this chapter delimit a possible flexible solution of the demonstrator.

With some enhancements the FLW files can be used to describe the scenarios, in stead of using a different format. The advantage of using the same files is the guarantee that the simulated scenario will *exactly* match the scenario described by an information flow diagram.

I have implemented a basic part of the demonstrator, but most credits go to Wim van Berchum who implemented the major part of the demonstrator. The demonstrator, which is suitable for stage 2 demonstrations in general, is described in detail in [17].

7 Conclusions and Recommendations

Broadband ISDN Signalling

A control system based on separation between CC and BC and the introduction of a more modular structure results in an improved flexibility which is essential for the second and third release of the broadband ISDN signalling. The new requirements to the signalling system originate from several sources, especially new telecommunication services.

The development of the new signalling system for B-ISDN release 2/3 is now focused on the second and third stage of the three stages methodology for characterization according to CCITT recommendation I.130. The second stage is divided into five steps that will be passed through several times.

At the beginning of this graduation task the state of the art of stage 2 comprises a functional model and a number of information flow diagrams for basic point-to-point calls with point-to-point connections. Basic call scenarios involving other call and connection types (p–mp, mp–p, mp–mp) must be developed, as well as scenarios involving other services.

Basic Functional Models

Several functional models are proposed for recommendation Q.73. For the description of multiparty scenarios the separated two level model is too restricted, so new – more complex – models are proposed. Up to now a very complex three level model (CC, RC, and BC levels) is accepted for MP cases. In the near future the three level model may be rejected, because it is too complex. The functional model with AC FEs has a real change to be accepted as the model to describe both point-to-point scenarios as well as point-to-multipoint scenarios. This model settles with questions like which FE on what level must take care of through-connection, allocation, and reservation.

The functionality of every FE is identified, though the list of functions may be incomplete. More important is the following: If the RC level will be rejected for some reason, the functions identified with the RC level must be rearranged in the CC and BC levels.

Information Flow Diagrams

The information flow diagrams describing point-to-point scenarios are quite stable, but other information flow diagrams describing MP scenarios are very unstable.

The release of bandwidth must be revised in order to create an unambiguous user plane state in the setup and release procedures.

The information flows described in stage 2 may not be mapped on stage 3 messages in advance. So implicit information flows must be explicitly described, although the information flows may be collected and sent as one single message in stage 3.

Several different call setup procedures exist. To support all procedures CC FEs need to be rather complex. Simultaneous setup and overlap sending are even unusable for MP configurations.

No new mechanisms are required for point-to-multipoint connections, but some existing establishment procedures cannot be used. Bi-directional connections cannot branch, hence a bridge is mandatory in this case. Some hardware implementations may not support branching, so not every node can be a branching node.

The RC level is responsible for the allocation of the bridge and the establishment of a bridge and all involving connections. If the RC level is present no direct relationships exist between the CC and BC level. In the three level model the BC FEs must be aware of MP configurations in order to control switch path through-connection and disconnection accurately.

The distributing section of a bridge may be seen as a separated branching function. This point of view excludes the existence of the MP–MP connection type, because the branching bearers must be uni-directional.

The AC FEs are the core of a node, containing shared functions and controlling through-connection of incoming and outgoing bearers. AC entities may be used as the interfacing point for IN purposes.

Within the AC model an important purpose of BC and CC FEs is internode information interchange. The BC level also controls all connections and the CC level controls all telecommunication services aspects of the basic call.

Most information flows for successful two-party basic call scenarios have been studied in detail. A number of mandatory and optional information items have been identified.

SDL Diagrams

The functional entities in the BC level are far less complex than FEs in the CC level. The CC FEs at the UNI require more functions than CC FEs at the NNI, because the UNI is less reliable.

Branching nodes are controlled by the incoming side of a node. Observe the IRBC and ORBC processes. The incoming BC process selects and instantiates the outgoing BC process. If the IRBC decides that user data has to be transported in several directions, several ORBC processes are started (one process per outgoing link). So the outgoing BC entities don't require branching functions. The incoming BC entities will control the user data copy features of the node.

The control of a bridge adds substantial complexity to the CC entities.

The major complexity in a CC process is caused by the existence of several setup procedures. The setup procedures differ quite a lot, so the process forks almost immediately after instantiation. The branches meet not before the setup phase has completed.

A reduction of complexity can be achieved by converting one setup procedure to another. This may be realized by IOCC. The IOCC entity will grow more complex, but all other CC entities in the network will get simpler, including the OCCA because this entity needs to support one procedure less.

In the AC model procedure conversion may be managed by AC entities, so the CC FEs may concentrate on controlling the call.

When an IRBC process receives a **BEARER_SETUP_req_ind** signal, the destination address must be inspected to determine which ORBC process must be instantiated (routing).
When the signal contains several destination addresses all addresses must be examined. For each destination an ORBC instance may be created, receiving a signal which contain only one address.
The differences between point-to-point and branching basic call scenarios seem to be diminutive.

In the SDL description the distributing section of the bridge is integrated as branching nodes. The O-LEX contains just the mixing section, called Mix Control Unit (MCU). The MCU is poorly specified in the diagram, so conclusions must be tread warily.
Controlling a bridge add a lot of complexity to the CC level. So the appellation 'Call Control' is slightly out of track. The three level functional model and the AC model preserve this inconsistency.

Q.73 Demonstrator

All requirements concerning flexibility can be met by using text files for the description of scenarios. Regarding the developments in CCITT, it is wise to describe the basic functional model in a text file too. Plain FLW files, as used by the program Flows, do not describe all the information required for simulation.
The user and technical requirements worked out in this report delimit a possible flexible solution of the demonstrator.
With some enhancements the FLW files can be used to describe the scenarios, in stead of using a new, incompatible format. The advantage of using the same files is the guarantee that the simulated scenario will exactly match the scenario described by an information flow diagram.
I have implemented a basic part of the demonstrator, but most credits go to Wim van Berchum who implemented the major part of the demonstrator. The demonstrator, which is suitable for stage 2 demonstrations in general, is described in detail in [17].

References

[1] CCITT WP XI/4-3, "Baseline text for the harmonized signalling requirements," *Temporary document XI/4*[1], Geneva, 9-21 to 10-2 1992.

[2] CCITT Recommendation Q.71, "ISDN 64 kbit/s circuit mode switched bearer services," Geneva, November 1988.

[3] CCITT Recommendation I.130, "Method for the characterization of telecommunication services supported by an ISDN and network capabilities of an ISDN," Melbourne, November 1988.

[4] CCITT WP XVIII/8, "Report of WP XVIII/8 ; General B-ISDN aspects," *Meeting report*, Geneva, June 11–28 1991.

[5] Vries, R.J.F. de, "Het ATM-concept ; een introductie," *Report 105 RNL/90*, Leidschendam, February 1990.

[6] ETSI SPS 3/DS, "Broadband integrated services digital network (B-ISDN) ; User-network interface (UNI) and network-network interface (NNI) application layer protocol for release 2/3," *Temporary document 09*, Stockholm, May 11–15 1992.

[7] CCITT SG XI, "Some basic issues for B-ISDN signalling protocol development," *Delayed contribution from Belgium*, Geneva, 9-21 to 10-2 1992.

[8] CCITT Draft recommendation Q.142X, "B-ISDN meta-signalling protocol," *Temporary document XI/4-40-E*, Geneva, March 10–17 1992.

[9] CCITT WP XI/4, "B-ISDN signalling group B report," *Temporary document XI/4-30*, Geneva, 9-21 to 10-2 1992.

[1] This document will result in CCITT Recommendation Q.73

[10] CCITT Recommendation Q.65, "Stage 2 of the method for the characterization of services supported by an ISDN," Geneva, 1989.

[11] CCITT Draft recommendation Q.73, "Basic call handling with separation of call control and bearer control," *Temporary document XI/4-13-E*, Geneva, September 21-30 1992.

[12] CCITT Recommendation I.362, "B-ISDN ATM adaptation layer (AAL) functional description," 1992.

[13] CCITT Draft recommendation Q.93B, "B-ISDN user–network interface layer 3 specification for basic call/bearer control," *Temporary document XI/6-37*, Geneva, March 9-20 1992.

[14] CCITT Recommendation I.361, "B-ISDN ATM layer specification," *SG XVIII SWP 8/1 Temporary document 66*, Geneva, June 1992.

[15] Gerrits, R.W., "Flows V2.0 user's manual," *Report NT-RA-92-1077*, 2nd edition, Leidschendam, November 1992.

[16] CCITT WP XI/4-3, "Baseline text for the harmonized signalling requirements," *Temporary document XI/4-E,* Melbourne, February 1-5 1993.

[17] Berchum, W. van, "Stage 2 demonstrator," *Report NT-??-93-????*, Leidschendam, June 1993.

[18] ISO/IEC DIS 9804.3, "Information processing systems – Open System Interconnection – Service definition for the Commitment, Concurrency and Recovery service element," *Draft international standard*, 1989.

[19] Belina, F. et al, "SDL with applications from protocol specifications," Prentice Hall, United Kingdom, 1991.

Acronyms

AAL	ATM Adaptation Layer
AC	Application Coordination
ATM	Asynchronous Transfer Mode
BC	Connection Control
BCA	Bearer Control Agent
B-ISDN	Broadband ISDN
C7	Signalling System 7
CBR	Continuous Bit Rate
CC	Call Control
CCA	Call Control Agent
CCITT	International Consultative Committee for Telephony and Telegraphy
CF	Control Function
conf.	confirmation
DAC	Digital-to-Analogue Converter
DSS1	Digital Signalling System 1
ETSI	European Telecommunication Standards Institute
FE	Functional Entity
HDTV	High Definition Television
ID	Identity
IN	Intelligent Network
ind.	indication
ISDN	Integrated Services Digital Network
LEX	Local Exchange
MCU	Mix Control Unit
MP	Multipoint / Multiparty
N-ISDN	Narrowband ISDN
NNI	Network-Node Interface
OO	Object-Oriented
OSI	Open System Interconnection
PE	Physical Entity
QOS	Quality of Service
RC	Special Resource Control
req.	request

resp.	response
SDL	Specification and Description Language
TE	Terminal Equipment
TEX	Transit Exchange
UNI	User-Network Interface
VBR	Variable Bit Rate
VC	Virtual Channel
VCI	Virtual Channel Identifier
VPCI	Virtual Path Connection Identifier
VPI	Virtual Path Identifier