# The interconnection of B-channel Packet Handlers in a SOPHO S network

Kok, W.

*Award date:*
1990

# The interconnection of B-channel Packet Handlers in a SOPHO S network

MASTER'S THESIS
by
W.Kok

Supervisor : Prof. Ir. M.P.J. Stevens (Technical University Eindhoven)
Advisors : Ir. E.L. van 't Hoff (PTDSN B.V., Hilversum)
Ir. M.J.M. van Weert (Technical University Eindhoven)

May 1990

# Abstract

The integration of a packetswitching service into the Sopho S, will enhance the capabilities of this switch. Apart from its already present circuitswitching functionality, a new service can be offered which is based on packetswitched data transport. For this purpose the B-channel Packet Handler (BPH) has been introduced. This BPH allows sets attaching to SOPHO S, to use the packetswitched infrastructure for communication via the D-channel. It offers a connection oriented service, which was primaly intended for set-server communication. This server can offer many applications to the users of Sopho S (name-number translation, directory enquiry etc.).

The BPH concept offers a single node packetswitching service which can serve no more than 240 sets. In order to be able to provide a more flexible service, it is investigated if networking with the B-channel Packet Handler is possible. Enlarging the BPH concept with networking capabilities, delivers a packetswitched architecture, which can be used for any application, based on packetswitched datatransfer.

The application provided by the server determines the grade of service which should be offered by the BPH network service. This can vary from low speed transaction oriented applications up to high speed bulk transfer of data. These functional requirements, demanded from the network service, result in the definition of the following :

1. Architectural requirements. A reliable infrastructure is needed to offer the BPH network service. Several topologies are suggested which serve best in different situations.

2. Definition of the protocol for interconnection. This protocol delivers a reliable means to transfer information accross the network. It is formally specified in the Specification and Description Language (SDL) of the CCITT. The correct working of this specification has been shown by means of simulation.

3. Network management. When offering a flexible network service, proper management is needed. Configuring the network and provisions which allow intervention in the operating of the network are important issues for network management. The methodolgy used by the standardization institutes (ISO/ECMA/CCITT) has also been used for the definition of the BPH network management. This allows easy migration towards standard management procedures.

It is shown that the B-channel Packet Handler concept is suited for offering a network-wide packetswitching service. Not only sets need to be served by the BPH network service. It is very well possible to use the packetswitching network service for dealing with the communication needs of the SOPHO switch itself (e.g. transferring important management information, downloading of set parameters).

The packetswitching network service concepts defined in this report, allow migration towards a fully integrated, distributed packetswitched network service. Apart from inintialization, this service is totally independent of the circuitswitching capabilities of the switch.

A new communication infrastructure can be realized with the B-channel Packet Handler network service, which can operate completely separated from the circuitswitched functionality of the SOPHO switch.

# Acknowledgements

I would like to thank the following persons for their support during my work at Philips Telecommunication Systems in Hilversum.

Prof. Ir. M.P.J. Stevens, for allowing me to graduate under his responsibility. M.J.M. van Weert, for his support on behalf of the University of Technology in Eindhoven. All the employees of BCS Low Range, for the nice conversations and the pleasant working environment they deliver. Special thanks to P.Blokland H.J.Donatz, and F.van Ormondt for supplying me with a comfortable room and lots of coffee. R. Ottink, for his support during the first three months of my working period. The contact person of STM (Software Tools and Methods), C. van Barreveld for his support with the SDL tools.

Last but certainly not least, special thanks to E.L. van 't Hoff, my coach at Philips, for his technical and social support, which helpt me to complete the project in a pleasant way.

# Contents

# List of Figures

# Chapter 1

# Introduction

In present office environments, communication takes place in a sophisticated manner. By means of a Private Automatic Branche eXchange (PABX), people are easier to reach, and this allows a more efficient working environment. Users of a PABX put great demands on the services offered by a telephone switch. The circuitswitched telephone service is the most important service offered by the PABX.

More functionality is expected however from a PABX. The ability to use the switch as a means to transfer data, is recognized to be very valuable. Therefore it is important to be able to offer an infrastructure which can realize this datatransport. A possible way of transporting this data, is by means of the already existing circuitswitched facilities.

Another way of transferring this data is by switching small packets accross a network to the other user. This way of transferrring information needs a so called Packetswitching Architecture. The data must be transformed in packets, and these packets realize the transport of data. Because this way of transferring data, allows a more efficient use of the communication channels, it is a very good alternative for data transport. Packetswitched data transport implies an architecture which is capable of dealing with packets. In the Synergetic Open PHilips Office Switch (Sopho S), a possible solution for offering a packetswitched architecture is found in the B-channel Packet Handler (BPH). With this BPH a packetswitched architecture is offered for the telephone sets attaching to the Sopho S. The primary goal of this B-channel Packet Handler is to serve as a packet router between the Sopho sets and an exterenal server. This server can offer many applications. For several reasons, the project of the BPH was cancelled.

Because packetswitching services become more and more important

nowadays, and the concept of the BPH is very applicable for offering these services, a new study is devoted to this subject. This study investigates the BPH concept and its networking capabilities. If the BPH concept can be enlarged with networking capabilities, a solution is offered for dealing with packetswitched services.

In the next section, the problem as it is formulated is depicted.

## 1.1 Formulation of the problem

The problem which is investigated in this report, is formulated as follows.

"... Integrating a packetswitching service into the Sopho S PABX, is to provide Sopho S with X.25 based capabilities in addition to its conventional circuitswitched functions. The primary objective of this packetswitching service is to provide a cost effective communication infrastructure between the proprietary 2B+D Sopho set terminals and an external server (e.g. a Telephone Management System).

To provide the communication capability needed to handle the packet data traffic between Sopho sets and external servers, without affecting the Sopho S performance, the concept of a so called B-channel Packet Handler has been introduced. The protocols defined are independent of the application and are designed to minimize the impact on existing hardware and software. Above all they allow a migration towards standards protocols such as the CCITT X.31.

In the current configuration, this packet handler can only serve upto 240 sets, and is only applicable for a single node network. In order to allow more than 240 sets to access the applications of a server, means must be provided to interconnect packet handlers.

For this purpose the following must be investigated.

- *Interlinking,* what requirements must be put on the network, and how can they be realized (e.g. topology).
- *Interworking,* what protocol should be used for the interconnection of BPHs.
- *Impact,* what is required from Sopho S (e.g.management).
- *Performance,* what is the performance of the network (e.g.throughput).

These investigations must result in a list of proposals. These proposals must be such that they can be used for further implementation ...."

## 1.2 Definitions

The following definitions are used throughout this report.

| | |
|---|---|
| **Set:** | Any Sopho S terminal which is capable of accessing the service of the B-channel Packet Handler. |
| **Server:** | Logical system offering an application to a user of the BPH service. This system does not necessarily have to be present in Sopho S. |
| **DLC:** | Digital Line Card capable of differentiating between several datalinks on the D-channel of the sets. |
| **User:** | This is either a set or server accessing the BPH service. |
| **Application:** | An application is offered by a server. |
| **Service:** | A service is offered by the BPH to be able to access an application. |

# Chapter 2

# Datacommunication and Sopho S

In this chapter a short introduction will be given into the concepts used for datacommunication. The rest of this report uses the specific terminology for datacommunication. Also the structure of a Sopho S switch will be treated.

## 2.1 The OSI model

### 2.1.1 Methodology used

In this section the ISO[1] reference model for Open System Interconnection will be explained. This model is developed in order to provide a framework which can be employed in the development of protocols for communicating devices. A protocol defines the rules and conventions used for this communication [3]. The ISO reference model is used by many organizations, which deal with the developing of protocols. The reason is that this model allows a structured approach for defining protocols. The concepts defined by this model will also be used throughout the rest of this report.

The OSI model consists of seven layers. Each layer performs a well defined function. The boundaries between these layers are chosen so that the information flow across the interfaces is as low as possible.

A layer consists of logical *entities*, which are the active elements in a layer, (e.g. a function, or group of functions). Two entities in the same layer, which are in separate systems, are called *peer-entities*. These peer-

[1] International Standardisation Organization

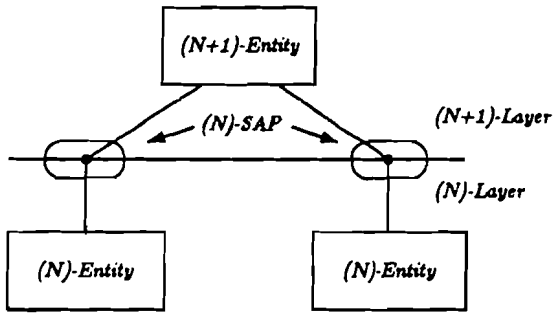



Figure 2.1: *The layered concept*

entities communicate by means of a peer-to-peer protocol. The (N)-layer offers services to the (N+1)-layer and uses the services offered by the (N-1)-layer. The point at which (N)-services are provided by an (N)-entity to an (N+1)-entity is called the *Service Access Points* (SAP's). Because a layer can have several entities, it can also have several SAP's (see figure 2.1). Inside a Service Access point there can be several *Connection Endpoints* (CEs), these are indicated in figure 2.1 with a •. By allowing several CEs in a SAP, multiplexing can be performed from layer (N+1) to layer (N) for one service. For information to be exchanged between two or more (N+1)-entities an association must be established between the entities in the (N)-layer using an *(N) peer-to-peer protocol.*

The data exchanged between the (N+1)-layer and the (N)-layer is called an (N+1)-PDU (*Protocol Data Unit*). For the (N)-layer this (N+1)-PDU is an (N)-SDU (*Service Data Unit*). The (N)-layer adds an (N)-PCI (*Protocol Control Information*) to form an (N) PDU (see figure 2.2). This protocol control information is needed for the correct operation of the peer to peer protocol.

Interaction between two adjacent layers is performed by means of *primitives.* A primitive may contain parameters. There are four different types of primitives (see figure 2.3).

1. *Request* primitive type, used when an (N+1)-layer is requesting service from an (N)-layer.

2. *Indication* primitive type, used by an (N)-layer to notify the (N+1)-layer of activities related to the primitive type request.

3. ***Response*** primitive type, used by an (N+1)-layer to acknowledge receipt of the primitive type indication (which came from the (N)-layer).

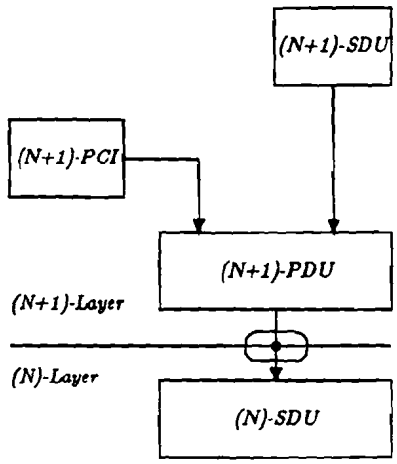



Figure 2.2: *The construction of an PDU*

4. *Confirm* primitive type, used by the (N)-layer providing the requested service, to confirm the (N+1) layer that the activity has been completed.

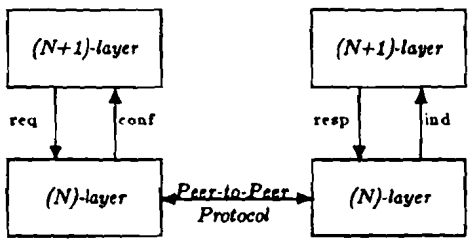



Figure 2.3: *The communication between layers*

### 2.1.2 The different layers

As described in the previous section, ISO uses a layered model to model the protocol stack of a communicating device. In this section a brief description will be given about the functions performed by the different layers.

**Layer 7 :** The Application layer.
This is the highest layer in the model. It provides services to the application processes which are also part of this layer. It

is up to the user to define the application present in this layer (e.g.database).

**Layer 6 :** The Presentation layer.
The purpose of this layer is to deal with the syntax and semantics of the data to be transmitted, independent of the application layer. Functions like encryption and compression are services offered by this layer.

**Layer 5 :** The session layer.
This layer manages the dialog in an orderly manner. It provides synchronization markers to indicate different communication phases. It also determines which service is allowed to send (allowing simplex, half duplex, full duplex).

**Layer 4 :** The transport layer.
This layer provides an end-to-end service to the upper layers. Functions like automatic recovery from loss of network connections, multiplexing from different transport connections on a network connection to reduce costs and additional error detection are performed in this layer.

**Layer 3 :** The network layer.
The network layer is mostly concerned with the routing of the datapackets through the network, from source to destination.

**Layer 2 :** The datalink layer.
The task of this layer is to transform the errorprone physical channel into a reliable link. Errors in frames are detected and possibly corrected. Also flow control and frame sequencing are functions performed by this layer.

**Layer 1 :** The physical layer.
This layer is concerned with the transmission of bits. Timing, voltages etc. are are dealt with in this layer.

## 2.2 Introduction into packetswitching

There are several ways in which a datacommunication service can be offered to a user. It can be very important in what way the datatransport takes place. Depending on the network capabilities and the user

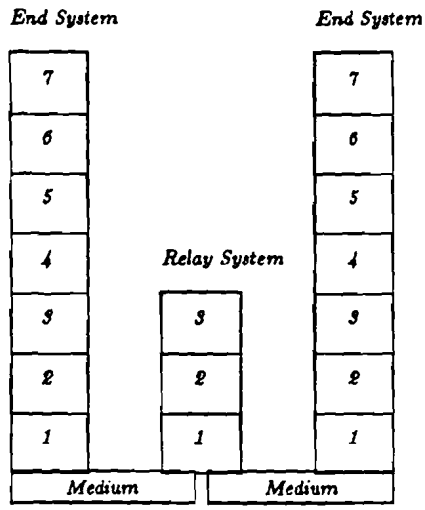



Figure 2.4: *The OSI reference model*

requirements, a certain service will be best. The following services can be offered.

**Circuit Switched Service :** In this case a physical path is established between the two users. This path can only be used by these users. This way of datatransport implies that the two users use the same physical interface and protocols, because no intervention of the network is present. A telephone call is a good example of a circuit switched service.

**Message Switched Service :** Here the network nodes are designed in such a way that they can store and forward a message from the user. A message can have any arbitrary length. In this way the channels used for communication, can be used by several users provided that the messages have been completely sent. The users may use different protocols to access this service. It is up to the nodes to perform the necessary protocol translation.

**Packet Switched Service :** With packetswitching there exists an upper limit to the length of the packets. This guarantees a higher throughput because packets from different users may be interleaved. This means that the channel can be used more efficiently. All the packets consists of a header, which takes care of the routing of the packet through the network, and a datafield which contains the information which must be transported.

The transport of data in a packetswitched network can be done in two different ways.

**Connection Oriented service :** When supporting a connection oriented service there are three distinct phases.

- Establishment Phase : This phase intends to establish a connection between the calling and called user. This connection is obtained through claims on resources in every intermediate node (processor-time, memory etc.). When this connection is established, an association exists between the two communicating entities, which allows reliable datatransfer.
- Data Transfer Phase : During this phase the data is transported. The way this is done is dependent on the protocol which is used for the datatransport.
- Release Phase : When there is no more data to be transported, the resources which were reserved by the user can be released so that they can be used by another user. This is done in the release phase by means of a special release command.

With a connection oriented service an association is established before datacommunication takes place. This means that during the actual data transfer, resources are guaranteed. This association is called a *virtual circuit*. This virtual circuit can be permanent or switched. In case of a permanent virtual circuit (PVC) the establishment phase is done only once on network initialization. During the lifetime of the network, the association between these two users is guaranteed.

With a Switched Virtual Circuit the establishment phase is performed every time data need to be transferred. When the transfer is completed the release phase is entered.

**Connectionless service** This service is a much simpler service than the previous one. It does not know the *establishment* and *release* phase but only a data transfer phase. Furthermore, no claims are made for resources in advance. This implies that a packet has to compete at every node with other packets arriving, to obtain the service. It is therefore possible that a packet must be discarded by a node, because the service cannot be offered at the time of arrival. Measures should be taken to prevent packets from being

lost at a node. The advantage of this service is that no resources are wasted by users who have no data to sent, but have a claim on them (as with PVCs). Every packet which is sent by the calling user must contain the destination address and the source address, so that the packet can be routed to the calling party and vica versa.

## 2.3 The Sopho switch

The Synergetic Open PHilips Office Switch (Sopho S) is a Private Automatic Branche eXchange, which offers different services to the user. The most important is the circuitswitched service, which takes care of the switching of telephone calls. This service offers a lot of facilities to the user, which enlarge the basic functions of a telephone set. Facilities like, calling number display, calls logged when absent, automatic ring back, follow me, are just a few examples of the many facilities which a Sopho switch can offer.

In figure 2.5 the functional decomposition of the Sopho S is shown.

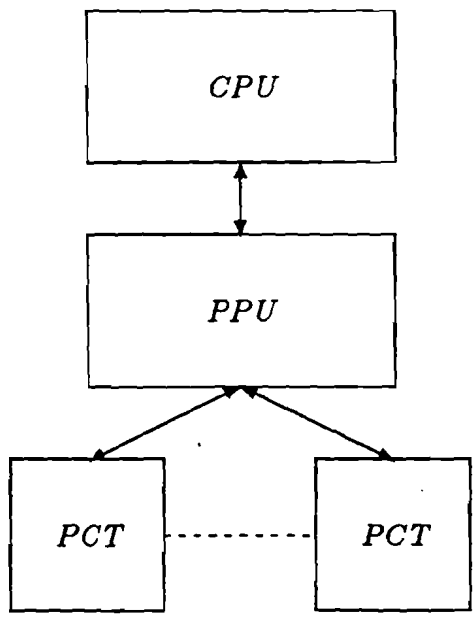



Figure 2.5: *Functional decomposition of the Sopho S*

**CPU** : The Central Processing Unit is responsible for controlling the

correct operation of the Sopho switch. Performing routing analyses, call control, system assurance, toll ticketing, are examples of tasks performed by the CPU.

**PPU** : The Peripheral Processing Unit releaves the CPU of many real-time functions. The CPU and PPU communicate by means of functional messages. The translation of these messages to physical signals for the PCT, is also an important task of the PPU.

**PCT** : The Peripheral Circuits allow different devices to connect to Sopho S. Sets can be attached by means of an Analog Line Card, or a Digital Line Card. For interconnection with the public net, an Analog Trunk Unit, and a Digital Trunk Unit exist.

The switching network, which is not shown in figure 2.5, supplies the connection between two users. The establishment of a connection is a task of the CPU.

This report deals with the B-channel Packet Handler. In chapter 5, the place of the BPH in the Sopho switch will be explained.

# Chapter 3

# Packetswitching

## 3.1 Existing packetswitching services

### 3.1.1 Introduction

In this section a study will be made for the state of the art of packetswitching services. This study is carried out for two reasons.

1. To get a good understanding of the packetswitching services present and their evolution to new standards. This knowledge can be used when defining the BPH network service.

2. To identify which packetswitching services are relevant for Sopho S to support, in the future.

Because of the fact that the Integrated Services Digital Network (ISDN) is seen as the network for the future, a lot of effort is being put in the development of a range of services which should satisfy the most appealing user needs for the coming decade. One service which the ISDN will eventually offer is the packetswitching service. Therefore an inventarisation of the most important developments of packetswitching in ISDN, will give a good overview of how packetswitching services will evolve.

The CCITT I. recommendations describe all aspects of the ISDN. Here only the elements of ISDN which are necessary to understand the rest of this section are described. The Integrated Services Digital Network supplies the user with a digital interface which offers many services (fax, telephone, telex etc.). ISDN offers, amongst other interfaces, a *Basic Rate Access* interface which consists of two B-channels and a D-channel. A B-channel is a 64 $kbit/s$ channel which can carry user

data. The D-channel is a 16 *kbits/s* channel which carries signalling data for the B-channel, but can also carry packetized user data. The protocols to be used on the D-channel for the three lower layers are standardized in the CCITT recommendations (Layer 1 : I.431, layer 2 : I.441/Q921 (=LAPD), layer 3 : I.451/Q931). In the ISDN there is a distinct separation between the *User-Plane* and the *Control-Plane*.
The protocols within the User-plane deal with the transfer of information among user applications. Any information which controls the exchange of data within a connection, but does not alter the state of the connection (e.g. flow-control), is also part of the U-plane.
The protocols which deal with the transfer of information for the control of user-plane connections, belong to the C-plane (e.g. establishing/clearing a connection, controlling the use of an established connection).

The X.25 protocol is a world wide accepted packet switching protocol, used to offer a packetswitching service to the user [72]. Because X.25 does not make a clear separation between the user and control plane, it does not nicely fit into the concept of ISDN. Therefore the need arises to adapt X.25 in order to fit in ISDN. It is clear that existing packet-terminals must be supported by the ISDN (certainly in the introduction phase of the ISDN), and that is why there is an evolution in the introduction of packet-switching in ISDN. In this way the ISDN can offer the same services which are already offered by the Public Switched Packet Data Network (PSPDN). Users are thus able to migrate to new services, while still keeping the old ones. CCITT defined four possible applications for these services :

- Block interactive data applications (low delay, high throughput)
- File transfer (delays not critical, higher throughputs)
- Multiplexed low bit rate (delay for study, higher throughput)
- Character-interactive traffic (low delay, low throughput)

Depending on the application, a different service should be offered by the network. As mentioned before there will be an evolution in offering these different services. The evolution of packetswitching services can be summarized as follows.

- Step 1 : X.31 services (section 3.1.2)
- Step 2 : Layer 2 multiplexing (section 3.1.3)

- Step 3 : Additional packet mode bearer services (section 3.1.4)

In the next sections these services will be dealt with.

### 3.1.2 Step 1, The X.31 Protocol

With the advent of ISDN in the 1980s, much investigation has been done to supply packet mode services in this environment. Because of the fact that X.25 had already become a worldwide accepted standard, X.25 has been standardized as a service that can be offered within an ISDN. This standard is defined in recommendation X.31. X.31 [4] uses in-band (no separation between U-plane and C-plane) rather than out-of-band (separation between U-plane and C-plane) procedures (the ISDN concept) for the control of virtual circuits. This approach is used to offer packet services with minimum deployment and interworking difficulties. For the communication over the B channel, the normal X.25 procedures can be used. The D channel which carries the signalling data need to be adapted however. The X.25 call control commandos must be transformed to LAPD signalling information.

X.31 specifies two integration scenarios:

**Access to PSPDN services** (formerly the minimum integration scenario) Here the ISDN provides a transparant or rate-adapted con-

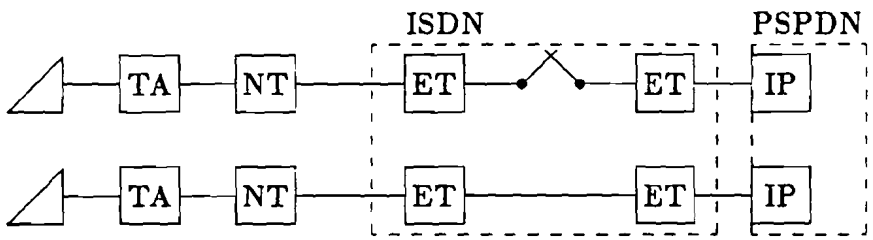



Figure 3.1: *Access to PSPDN services*

nection for an X.25-based terminal to/from a PSPDN port. Normal ISDN (I.451) signalling procedures are used over the ISDN to establish the connection of a B-channel to/from the access unit of the PSPDN. X.25 call set-up procedures are then used over this B-channel to establish the connection with the PSPDN. When setting up the B channel the address of the IP is needed (ISDN address). The X.25 call setup procedure needs to know the address of the called DTE. So two addresses are needed (see figure 3.1).

**ISDN virtual circuit bearer service** (formerly the maximum integration scenario). In this case the service is provided within the

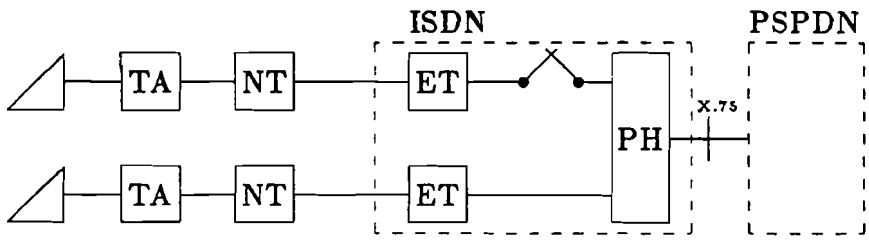



Figure 3.2: *ISDN virtual circuit bearer service*

ISDN. The user protocol is terminated by a Packet Handler (PH) rather than the PSPDN. The PH interfaces to the PSPDN via the X.75 protocol (Fig. 3.2). Access to the PH can be via the B or the D channel, which is determined by either the ISDN or the called DTE. The ISDN signalling procedures does not need an address (by asking for a packetswitching service, the PH is automatically selected). The X.25 call setup procedure needs to know the address of the called terminal. So here only one address is needed.

- Access via the B channel
  In this case the ISDN will establish a circuit switched connection to the PH on receipt of a 'packet transfer mode' request, by use of the D channel protocol. Once this connection is established the packet-mode terminal can use the X.25 protocol over the B-channel.
- Access via the D-channel
  In this case a distinction has to be made between normal signalling data and X.25 packet data. This distinction is made by means of a Service Access Point Identifier (SAPI) in the frame header. The network will route the packet data frames to the PH, the signalling frames will be dealt with as before. The D-channel thus allows the interleaving of packets of data and signalling data.

What X.31 does is really just the encapsulation of existing X.25 packet-mode services in an ISDN environment, at the physical interface. In both type of services (*access to PSPDN* and *ISDN virtual circuit*

*bearer service*) existing in-band control procedures in X.25 continue to be employed for the establishment and release of virtual circuits. Thus the complete packet layer of X.25 is used for Call Control and the Data Transfer Phase over the ISDN.

### 3.1.3 Step 2, Layer 2 Multiplexing

This step proposes the use of LAPD not only over the D-channel, but also over the B-channel. Multiplexing in X.25 is achieved through the use of logical channels in the packet layer (layer 3 multiplexing). When using LAPD over the B- channel, multiplexing is done at layer 2 through the statistical multiplexing of different Data Link Connections (DLCs) on the same physical channel. These DLCs are identified by different Data Link Connection Identifiers (DLCIs). A DLCI consists of a Service Access Point Identifier and a Connection Endpoint Identifier. These DLCIs have local significance, their value is only unique in a given physical channel. Switching in the link-layer is achieved by binding the DLCIs to routing info at the intermediate nodes to form a set of network edge-to-edge logical paths (thus a connection-oriented service may be offered). At layer 3 the X.25 packet layer exists. Layer 3 and layer 2 multiplexing may exist simultaneously.

### 3.1.4 Step 3, Additional Packet Mode Bearer Services

The next step in the evolution of packet switching in the ISDN is the separation of the user and control plane. By separating the user and control plane, there is no need anymore to fully terminate the protocol in the user plane. To come to this form of packet switching, four types of additional packet-mode services are defined in recommendation I.122[1]. All these services adopt out-of-band signalling. The services are [22] :

1. frame relaying 1
2. frame relaying 2
3. frame switching
4. X.25-based additional packet-mode

[1]Not all these services will be standardized however. Studies are going on which services are superfluous

The user-network interface protocol reference model is show in the figure below (Fig. 3.3).

For the control plane I.430/431 provides the layer 1 protocol, I.441

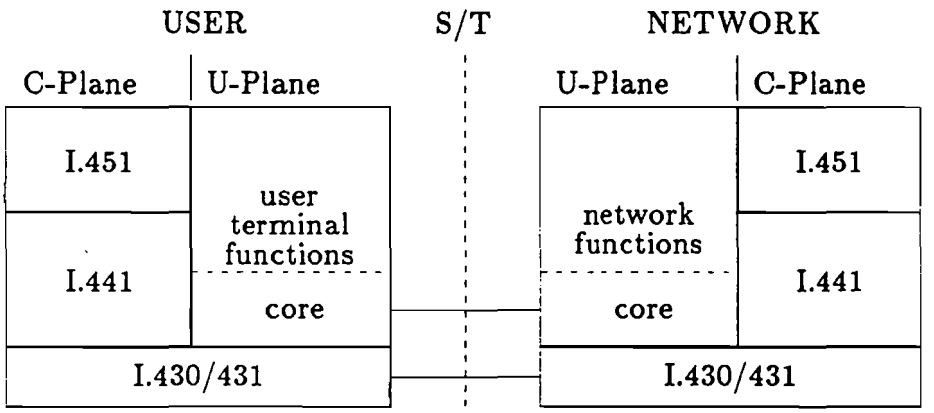



Figure 3.3: *User-network interface reference model*

and I.451 provide the layer 2, respectively the layer 3 protocol, both on the user and network side (over the D-channel). For the user plane the layer 1 protocol is the same, but the layer 2 and 3 protocols are different depending on the service provided. If a service is requested by the user which needs low delay and high throughput, the network may choose a service which does not fully terminate layer 3 protocol handling. Instead a service may be chosen by the network, which only handles layer 2 functions (and thus for example discarding a wrong frame instead of waiting for retransmission).

In figure 3.4 the services and the protocol terminations are summarized.

To lower the delay, some of the services only provide the core functions of the I.144* protocol. These core functions of I.441* (which is I.441 with appropriate extensions) are:

- frame delimiting, alignment and transparency.
- frame multiplexing/demultiplexing using the address field.
- inspection of frame length.
- detection of transmission errors.

| Bearer Service | User Side | Network Side |
|---|---|---|
| frame relaying 1 | I.441* core | I.441* core |
| frame relaying 2 | I.441* | I.441* core |
| frame switching | I.441* | I.441* |
| X.25-based p-m | I.441* and X.25 dtp | I.441* and X.25 dtp |

note : dtp = Data Transfer Part

Figure 3.4: *Protocol termination for different services*

The different services for the user plane will be pointed out in the next sections.

### 3.1.5 Frame relaying 1

This type of service uses less processing in the network and thus leads to shorter delays. Also more users and a higher volume of traffic can make use of the network when this service is provided. Because of the lower layer protocol termination of the network, the user has to do more processing by himself (e.g.ask for retransmission etc.) The frame sequencing and the error control is end-to-end. The U-plane of this service is shown below (Fig. 3.5).

If a frame is detected at the S/T interface which does not conform to

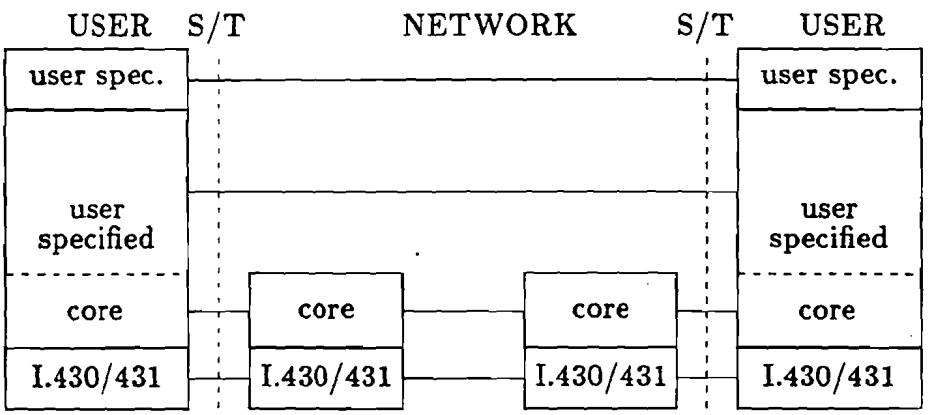



Figure 3.5: *Frame relaying 1 service, U-Plane*

I.441* core formats it is discarded. It is up to the users higher layer protocols to detect frame loss and ask for retransmission. In all other

cases the frame is relayed to another node according to the routing tables.

### 3.1.6 Frame relaying 2

The U-plane of this service is shown below (Fig. 3.6).

The terminals operate end-to-end with the complete I.441* protocol.

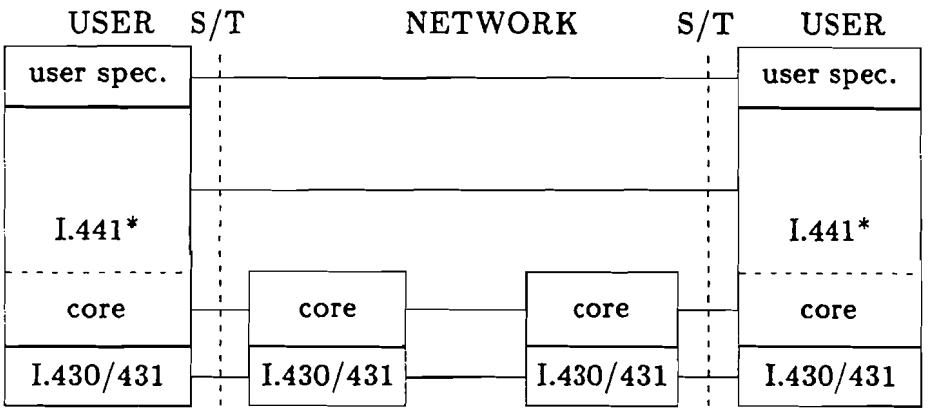



Figure 3.6: *Frame relaying 2 service, U-Plane*

The same remarks apply as for frame relaying 1.

### 3.1.7 Frame switching

This service increases the processing time within the network (because it supplies a frame-sequenced, error-free environment). As a result of this the users don't need to deal with these tasks themselves. The U-plane of this service is shown below. The bearer service supplied by the network supports the full I.441* function (Fig. 3.7).

### 3.1.8 X.25-based additional packet mode

The main reason for providing this kind of service, is to support OSI applications provided on the user devices. Full layer 3 protocol termination is provided by the network. The U-plane of this service is shown below (Fig. 3.8).

At layer 3 the data transfer phase of X.25 will be used. Most of the call

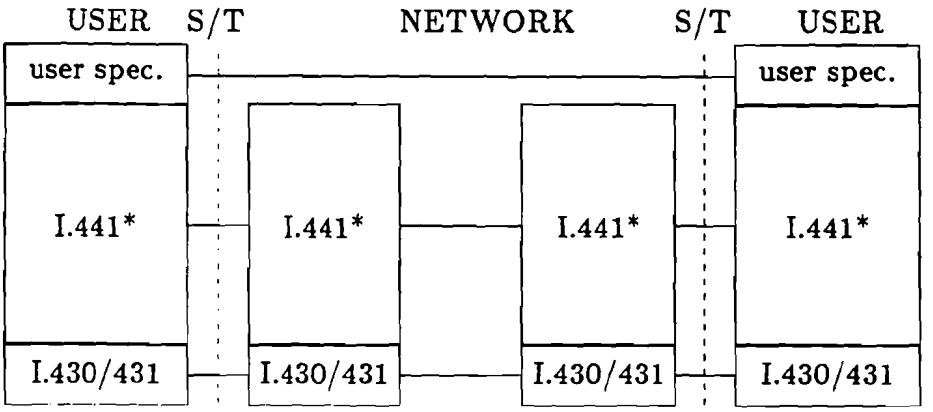



Figure 3.7: *Frame switching service, U-Plane*

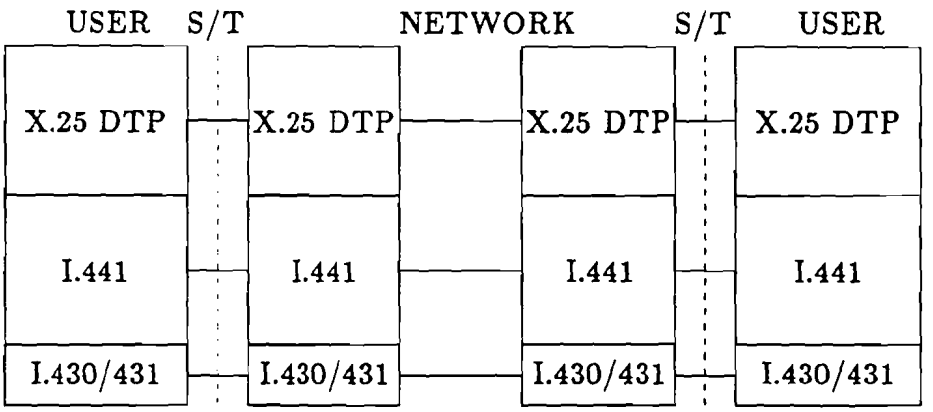



Figure 3.8: *X.25-based additional packet mode service, U-Plane*

control procedures of X.25 will not be needed because these functions are done by the I.451 protocol over the D-channel.

### 3.1.9 Concluding

Why should this evolution take place ? Why don't stop after step 1 (X.31) has been provided. The main motivations for this evolution can be classified into two categories.

- Technological
  The ISDN already uses layer 2 multiplexing on the D-channel, so why not use it on the B-channel too.

The circuit-switched call control in ISDN is already based on out-of-band signalling over the D-channel. It is attractive to use the same signalling for packet-mode services.

- New Applications
  New applications are being developed (e.g. packetized voice and data). The use of one signalling protocol could greatly reduce the costs for such applications.
  New applications might need higher throughput, this can be achieved by lower layer protocol termination within the network.

- X.25 network will become obsolete with the advent of the ISDN.

The evolution described in this paper and the new bearer services, are still investigated by the CCITT. Probably not all which is described here will be provided in the future. During the study period 88-92 of the CCITT, decisions will be made which services are going to be provided.

## 3.2 Present development of packetswitching in Sopho S

Sopho S also acknowledged the need to be able to support packetswitching services. In this section a description will be given of the packetswitching services which are defined for Sopho S. Not all these services are supported yet, but it is recognized that most services will need to be supported in the future.

**Packet server for Sopho S :** This study [6], defines how a packetswitching service based on the X.31 Access to PSPDN services[2], can be realized in a Sopho S environment. The purpose is to enlarge the Sopho S with packetswitching functionality. The packet server allows packetswitched based terminals to communicate with each other. This communication can take place locally, that is in a Sopho switch, but it is also possible that the terminal communicates with a terminal across the PSPDN or ISDN (when Sopho S interfaces with these networks of course). The packet server shall use the Sopho S B-channels for building communication paths from terminal to terminal. Once this channel has been established normal X.25 can be committed. This solution does not offer the capability to use the D-channel for packetswitched datatransfer.

[2]See section 3.1.2

**User to user signalling :** With the existing equipment it is possible to use the D- channel of the sets to transfer packetized data. Along with the call-setup message, which is used to establish a call with another user, a datapacket can be sent. The drawback of this method of packetswitching in the D-channel is that it needs CPU intervention. For small message transfer between users this does not have to be a problem. When this capability of the D-channel is to be used to transfer more information, the loading of the CPU may become too large.

**Internal communication network :** To overcome the problem of the loading of the CPU another communication path in the Sopho S is established [5]. This path bypasses the CPU and is used to transfer user data. Only during the setup of this establishment CPU intervention is needed. When the addresses of both communicating users are known, the datatransfer can take place without any further intervention of the CPU. This new communication structure could also be used for the transfer of packetized data.

**B-channel Packet Handler :** The B-channel Packet Handler[3] is a board which was specially designed to enlarge Sopho S with packetswitching capabilities [25]. Apart from initialization, no intervention of the CPU is needed. The sets access the BPH via the D-channel. This means that a separation should be made between normal signalling information and packetized data. This will be a task of the Digital Line Circuit (to which the sets are connected), which will use layer 2 multiplexing to separate between these two communication flows. The main objective for the BPH is to serve as a packet router between the sets and a server. The server can have several applications like name-number translation or electronic mail. The BPH will be treated thoroughly throughout the rest of this report.

## 3.3 Sopho S and future packetswitching services

Taking into account the current development of packetswitching services in ISDN, it can be important for Sopho S, to be able to support such

[3]The name of the B-channel Packet Handler is rather misleading. The sets apply for the service via the D-channel, so DPH seems a more logical name.

services. A proprietary solution for the X.31 Access to PSPDN service is already defined [6] . For the X.31 ISDN virtual circuit bearer service, it must be possible to access the services also via the D-channel. The BPH is a good alternative to offer this service. It is designed to interface with the sets via the D-channel, when appropriate measures are taken for the Digital Line Circuit. If the BPH terminates the ISDN protocols of the sets, and offers an X.75 (or proprietary) service towards the packetswitched network, no special hardware has to be designed to offer this service. Note that this application of the BPH is not what it is intended for. Throughout the rest of this report therefore, the BPH with its original applications is dealt with. The problem with the BPH is that it is not able to have B-channel access, so a special solution should be taken here (mixture of packet server and BPH). It should be investigated if there is the need by customers of an ISPBX (ISDN PABX), to be able to use the additional packet mode services as described in recommendation I.122. Users may demand a reliable transfer of their datatraffic so the X.25 based additional packet mode service seems rather important to be supplied by the switch. If the customers do their own end to end control of the datatraffic it can be a wish that the network does not intervene to such a high level. Therefore also the lower protocol terminations may be important to support.

Because no market study was performed it is impossible to advise which services should or should not be supported. It is important however to follow the developments of the packetswitching services, so that services can be offered when the need arises.

# Chapter 4

# The B-channel packet handler

## 4.1 Introduction

As was shown in chapter 4, there are several ways of delivering a packetswitched service. The B-channel packet handler is a means to offer such service. The primary objective of this packetswitched service is to provide a cost-effective infrastructure between a Sopho-set and a server. This server can be used as a centralized directory or messaging server. To be able to support this service, a B-channel packet handler (BPH) has been developed. For several reasons the BPH has not yet been implemented. The board has been designed, and produced, but no further effort has been put in implementing the BPH. The reason for developing a special board for this service was to minimize the loading of the Central Processing Unit, thus the board takes care of the datatraffic between set and server, without the intervention of the CPU. The primary application of the BPH is to serve as a packet router between a server and the Sopho sets. The following applications are the most appealing application which could be offered with the BPH concept [1] [26].

**Directory Enquiry :** Via the BPH it should be possible to do an enquiry in the server (e.g. information about a person).

**Name Dialling :** The name-number relation is stored in the server. A centralized database has advantages over a local database in the set, in that it is easier to maintain.

[1] Of course many more applications than the ones outlined here are possible to offer with the BPH.

**Calling Name Display :** Not only the number but also the name and other information of the calling user, can be displayed.

**Electronic Mail :** *It* is possible to leave a message for a certain person . These messages can be retrieved by the called persons on demand.

All these facilities involve the exchange of packets between set and server. The Sopho sets communicate with the BPH via the D channel. Therefore the D- channel must allow the interleaving of signalling-data with packet-data. The separation of packet-data and signalling-data is a task of the Digital Line Circuit (see figure 4.1).

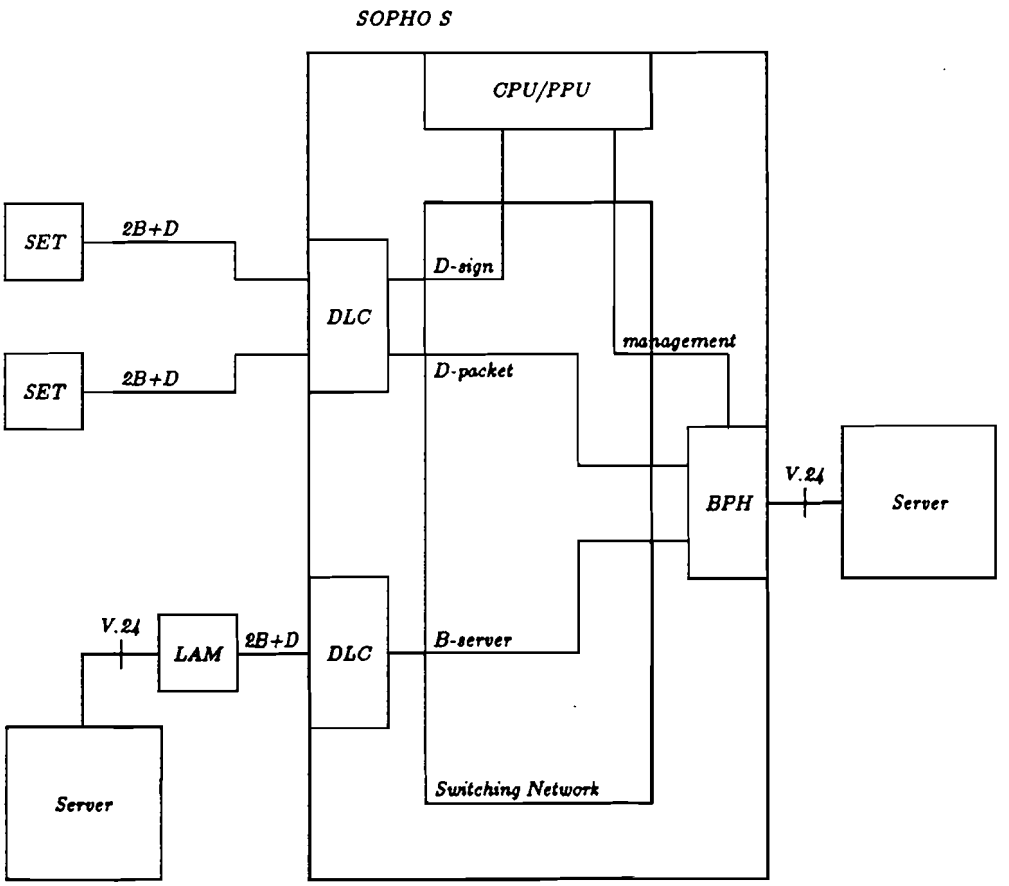



Figure 4.1: *Sopho S packetswitching architecture*

The BPH can handle four external servers and a maximum of 240 Sopho sets. The servers can attache to the BPH via a V.24 cable or via a circuit-switched B-channel. Although primary designed for set-server applications, the BPH is capable of dealing with all kinds of packetswitched services in the D-channel (see chapter 4).

When the BPH was designed a lot of requirements had to be met. Amongst others the BPH-design had to meet the following requirements. It should [26]

1. have minimal impact on the existing Sopho S hardware and software,
2. minimize the effect on Sohpo S circuit switching facilities (PPU and CPU),
3. allow all different Sopho sets, to access the packetswitched service,
4. minimize the impact on the server (TMS system), meaning among others that the protocols used by this server should be taken over,
5. provide an architecture that is extensible for future evolutions.

So it should be able with the BPH to interwork with all existing equipment needed to offer the packetswitched service, while minimizing the impact on Sopho S circuitswitched facilities.

As the BPH is primary designed to serve as a packetrouter between a Sopho set and a TMS server, the protocols needed for this communication are already defined.

## 4.2 Current BPH services/concepts

The approach used for providing the packetswitched service outlined in the previous section, is through establishing a Permanent Virtual Circuits between every communicating pair. This PVC is a permanent association existing between two endpoints which requires no call set-up or call clearing actions. It represents a flow controlled transport mechanism which is independent of the application it is used for. The reason for using Permanent Virtual Circuits in stead of Switched Virtual Circuits is that the former are much easier to implement. A migration to Switched Virtual Circuits is possible. Because these SVCs will need Packet Switching Call Control, this is a much more sophisticated solution for providing a packetswitched service.

Each packet transmitted over a PVC carries a Logical Channel Number (LCN), which identifies its destination. Using this approach implies that no real CPU intervention is needed, except that the CPU will have to provide the BPH with the necessary operational information, during initialization. The sets access the server by means of this LCN. So each Sopho set contains one LCN per server application it may access. The server also uses LCNs for addressing the sets.

The protocols defined so far to realize the BPH service, are depicted in figure 4.2. As can be seen from figure 4.2 the BPH only supports the

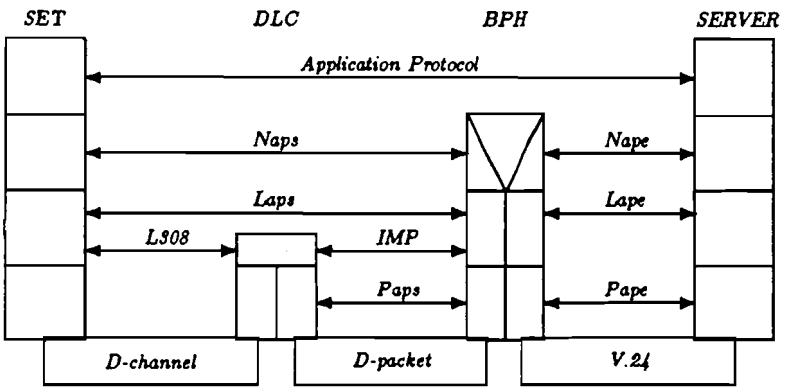



Figure 4.2: *Protocol Stack for packetswitching*

three lower layers. A datalink protocol is defined (Laps [23]) on top of the proprietary lower layer 2 LAM308 protocol. This protocol is used by the sets to connect to the Sopho switch. Laps is based on a subset of the CCITT Lapd (I.440/I.441) protocol and allows multi datalink connections to be transported over one D-channel. This allows for one datalink carrying signalling data for normal set operation, and possibly several datalinks for carrying packet data. The separation of packet data and signalling data is performed by means of different Service Access Point Identifiers.

1. SAPI 0 is used to denote the signalling messages. In this case the datalink protocol (not Laps) is terminated by the DLC.

2. SAPI 16 is used to denote packet data. The BPH takes care of the correct handling of the datalink protocol. Datapackets are wrapped up by the DLC in Inter Message Protocol (IMP) frames and relayed to the BPH via the D-packet channel. The IMP pro-

tocol is a proprietary Sopho S protocol used for the exchange of messages between control units and intelligent line cards.

The Naps protocol [27] is a subset of X.25, and will as a start only deal with Permanent Virtual Circuits.

At the server-side the proprietary Business Communication Systems (BCS) protocol is used for layer 2 communication. This protocol is developed for a Telephone Management System (TMS). To minimize the effort in developing a new server, the TMS was chosen as a solution for implementing a server. This choice implies that the protocols defined for this server, also have to be used at the BPH side. For future applications the BCS protocol will possibly be replaced by a HDLC-like protocol, in order to migrate to standard protocols.

Nape [24] is also a subset of X.25 and is much like Naps. The main difference between these two protocols, is how they interface with layer 2.

## 4.3 The desired situation

The BPH as it is designed has some limitations which can become annoying when a more flexible service should be offered. The current BPH concept does not support end-to-end communication between arbitrary users and does not have any network provisions. The former requires an upgrade of the network access protocols, and is outside the scope of this report. Extending the BPH with networking capabilities, has the following advantages.

1. More than 240 sets can be served.

2. Sopho sets in different switches can access the application offered through a BPH in another switch.

3. A packetswitched network is offered, which can be used for many different applications, concerning data transport between different Sopho switches.

The requirement to allow networking of BPHs throws up a lot of problems, which need to be solved.

1. What requirements does the network service has to meet, and how can these be realized with the BPH concept.

2. What kind of protocol should be used to offer the network service.
3. How can the network be maintained. This means that network management should have the means to perform actions on the network.

These problems will be dealt with in the rest of this report.

# Chapter 5

# BPH network aspects

## 5.1 Network requirements

For several reasons it is attractive to extend the B-channel packet handler with networking capabilities. A BPH is designed to serve a maximum of 240 sets. When more than this maximum number of sets want to make use of the service offered by the BPH, a problem arises. More sets can be served however when it is possible to interconnect the BPHs. Another reason for interconnecting BPHs can be that users of a Sopho

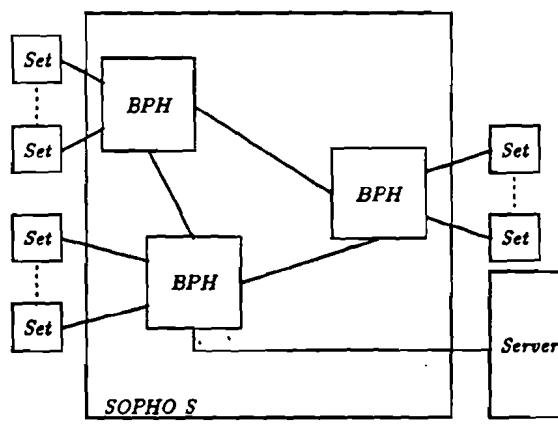



Figure 5.1: *Network of BPHs in one Sopho switch*

switch, want to access an application not present in their own switch. With a network of BPHs, a packetswitched architecture is present in the SOPHO S environment, which can be used to access different packetswitched applications.

Two distinct situations are present when the BPH is used for offering a network service.

1. Network of BPHs in one Sopho Switch. This situation arises when more than the maximum number of users want to access the BPH service (figure 5.1).

2. Network of BPHs distributed over several Sopho switches (figure 5.2). This situation occurs when users of the BPH service are distributed over several Sopho switches, the BPHs do not necessarily need to serve the maximum number of users.

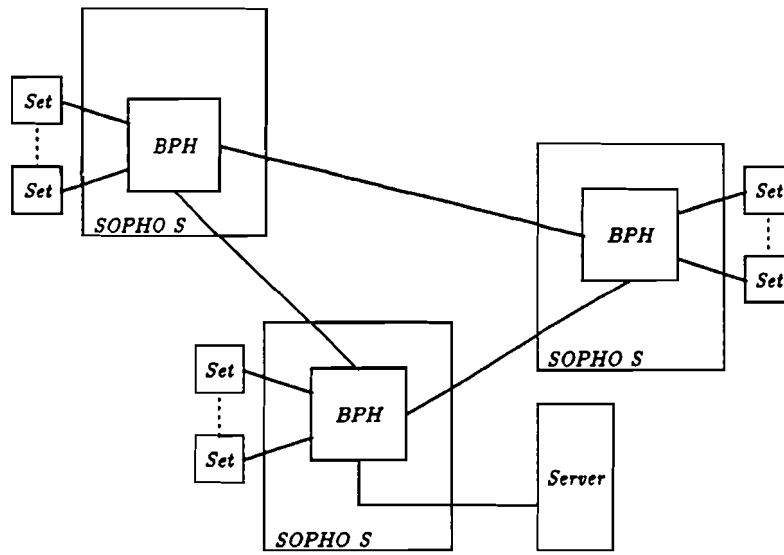



Figure 5.2: *Network of BPHs divided over several Sopho switches*

When networking of BPHs is possible, a more flexible service can be offered with the B-channel Packet Handler. This service allows more users on different physical locations, to access the application. Of course this network is only valuable, when it can meet the requirements, put on the network service. The following requirements can be identified.

**Functional requirements** : These are the requirements as seen from a user point of view. The user is not concerned with the architecture to meet these requirements.

1. The network of BPHs should not influence the protocols, used for the network-user communication. It must allow these protocols to migrate to future standards.

2. The network of BPHs should have no influence on the application offered by the application server.
3. New network users should be able to access the network without any limitations on the physical location.
4. The application accessed by the user should experience an acceptable delay from any physical location in the network.
5. The network should offer a reliable means for accessing the different server applications.

**Architectural requirements** : These are the requirements the network must come up to, in order to be able to meet the functional requirements mentioned above.

1. Removal, addition or failure of a BPH should not disturb the normal operation of the network. At least part of the network should be able to operate in case of a failure.
2. Architecture should allow a fair distribution of delays. This implies a symmetrical architecture.
3. The costs for adding a new BPH to the network should be minimal. When distributing the network over several Sopho switches, the number of interconnections should be kept as low as possible. Leased digital lines will be used for these interconnections, which are normally rather expensive.
4. The network should be configured such, that it can offer a low delay and reliable service.

Seen from the service provider point of view, single-node BPH service differs a great deal from the service provided by a network of BPHs. The following problems should be solved in order to define a network service, which can offer the same service as a single BPH.

1. Will the interconnection service offer a connection-oriented or a connectionless means for transferring user information ?
2. How to obtain a unique addressing scheme, which is applicable for the entire network ? Each user needs to be addressed in order to provide a meaningful service.
3. What architecture serves best, to meet the functional requirements ? This should result in the definition of the topology of the network.

These problems will be dealt with in the next sections.

## 5.2 The interconnection service

The single-node BPH concept offers a connection oriented service for the transfer of user information [25]. For every communicating layer 3 entity of a user, the BPH contains a peer entity. These associations are on a permanent basis, meaning that during initialization of the BPH the resources are claimed and the different relations are established.

When considering a network of BPHs it is not economical anymore to offer a connection oriented service based on permanent virtual circuits. Consider a network as depicted in figure 5.3. With this network it is

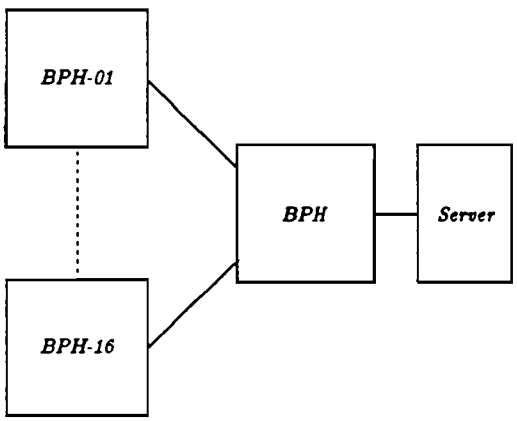



Figure 5.3: *Network with three BPHs*

possible to serve a total of 3600[1] sets. All these sets have two ports, so even for this very simple network 7200 permanent associations are required. It is a waste of resources to reserve them in advance because most of the time they are not needed. It is much more effective to claim the resources when they are needed. Two possible solutions are considered.

**Switched Virtual Circuit** : With this approach a connection oriented service is offered by the network. When information has to be transferred, a BPH-node establishes a connection with its peer, before information transfer takes place. When the information has been transferred the resources can be released.

Due to the short length of the information packets, the overhead of establishing and releasing associations, is not considered valid.

[1] 16 BPHs, 15 DLCs per BPH, 15 sets per DLC. See also section 5.3.1

This implies too much a degradation of the performance.

**Connectionless Approach :** With this approach no association exists between two communicating BPHs. The information is sent to the peer-BPH, without first establishing a connection. It is possible that the peer BPH is not able to serve the packet at the time of arrival (due to the lack of resources). Proper measures should be taken by the protocols which realize this service, to prevent information from being lost in the network.

Because of the arguments mentioned above the choice is made to use the connectionless approach for interconnecting BPHs. In chapter 7 the protocol for interconnection of BPHs is described.

## 5.3 Addressing the BPH

When offering a network service with the BPH, the problem arises that each user of this service, needs to be addressed in a unique way. Only then, the information exchange between two users can be realized. Therefore an addressing scheme has to be defined which uniquely identifies each user.

### 5.3.1 The old situation

In figure 5.4 it is shown how the configuration with one BPH looks like. A BPH cannot support more than 16 Digital Line Circuits. Each DLC can connect 15 sets, so a total of 240 sets can be attached to a BPH. The DLC is transparent for packet data to/from users. For addressing purposes however it is not transparent. The BPH uses a unique link address to access each B-channel of a set individually. These B-channels are identified by a port number. The link address is of the following format [28].

$$link\ address\ = <\ DLC\#,\ line\#,\ port\#\ >$$

This link address is assembled during the transport of the packet. The DLC adds the line# (1..15) and the port# (0..1). Finally the BPH adds the DLC# (0..15) to complete the linkaddress. The BPH maps these link addresses onto a logical channel number (lcn). This results in 480 different lcns, which are used for addressing the users. This information

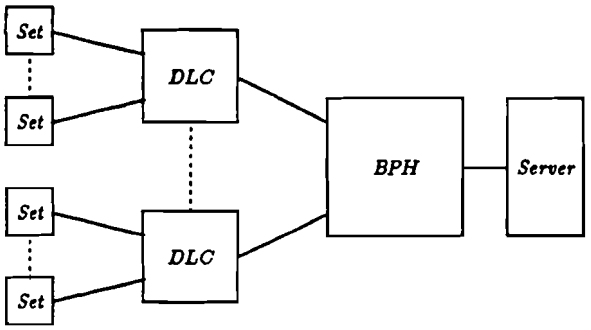



Figure 5.4: *Configuration with one BPH*

uniquely identifies each user of the single-node BPH service. The BPH uses the linkaddress to address the right layer 3 entity in the BPH.

To access the server the sets use a logical channel number to identify the application. Because the decision was made to use permanent virtual circuits for set-BPH communication, all these lcns are assigned when installing the BPH. The BPH should contain a routing-table which holds the relation between the different lcns and the linkaddresses.

### 5.3.2 The new situation

When interconnecting BPHs, it must still be possible to address each user separately, in order to support the services offered by a BPH. Consider the network of figure 5.5. Two separated addressing domains are defined (see figure 5.5).

1. The *Local Addressing Domain.* This domain is concerned with the addressing of the BPHs. In this domain no users are present, just BPHs.

2. The *BPH Addressing Domain.* This is the domain which is concerned with the addressing to a user when the packet has already been routed via the local domain to the right BPH domain.

The reason for splitting the network into two domains is that the problem of interconnecting BPHs must not influence the protocols which are already defined for the BPH-user communication. Therefore the old principles apply to the BPH addressing domain, and an addressing method must be derived for the local addressing domain.

Offering a network service implies the transport of information between the different BPH domains, via the local domain. Therefore the

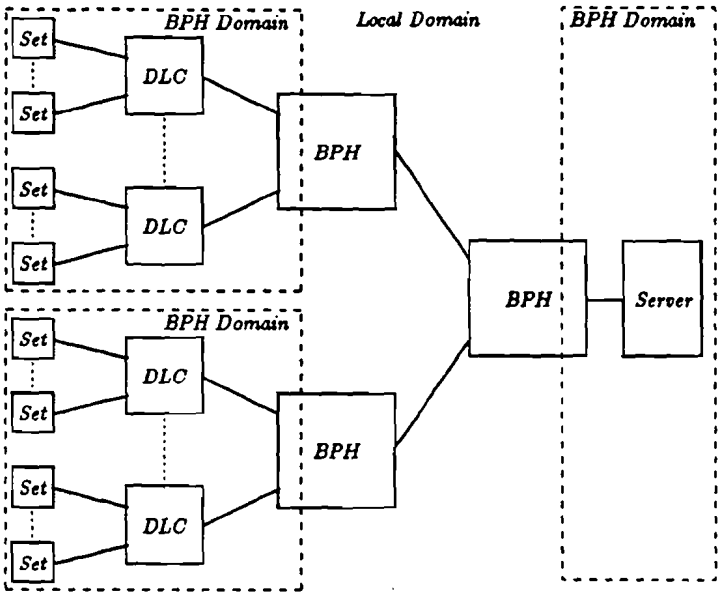



Figure 5.5: *Network with three BPHs*

addresses to be used to identify users, should be unique in the total network. With the addressing principles derived so far this is not the case. The addresses of the users are only unique in their BPH domain. Interconnecting these BPH domains results in addresses which are no longer unique. There are several possibilities to obtain a unique address, which will be treated next.

**Sustaining the same principle :** When using the same principle for addressing as was shown in section 5.3.1, it is possible to obtain an address which is unique in the entire network. The BPH which receives a packet adds the number of the BPH (0..15) to the address information received. The address has the following format.

$$link\ address = < BPH\#,\ DLC\#,\ line\#,\ port\# >$$

This way of adding address information can be a recursive process at each BPH in the path from the packet from set to server. When the packet arrives at the BPH with the server, the total address is obtained. This address can now be used to route the packet via the local domain to the right BPH domain, and finally to the right user.

This way of addressing has some drawbacks however.

1. The address determines the route which the packet has to take, therefore it cannot be used for alternative routing strategies.
2. The length of the address relates to the physical place of the set in the network. Depending on how many hops have to be taken, the length of the address becomes larger.
3. This way of addressing is very messy. Every BPH needs to intervene at a rather low level (which port does the packet come from) to add some address information. It is very dependent on the architecture of the BPH, it is not an open-ended solution.

To overcome these drawbacks, another addressing method is studied.

**Absolute addressing :** With this method of addressing each BPH is assigned an address which is unique in the total network. This address can then be used for routing in the local addressing domain. When the packet arrives at the right BPH domain the addressing information for this domain can be used for further routing. Because a packet can be sent through the network, the BPH domain addressing information must also be part of the total addressing information. Only then it is possible to route the packet to the right user. These addresses of the BPH can be seen as layer 3 network addresses. It will be the task of the layer 3 of the protocol for interconnection to take care of the correct routing of the packets through the network (see chapter 7).

The second way of addressing BPHs is much more structured and better to use. That is why this method will be used as addressing scheme for the network service. In figure 5.6 the address information needed is shown. The first field contains the address of the BPH domain which

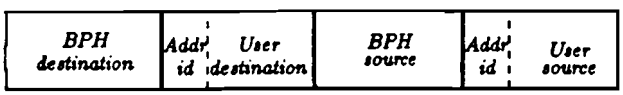



Figure 5.6: *The address information*

must be reached and is used to route the packet to the right BPH domain (Local addressing information). When arrived at the BPH the

second field determines which user should be accessed (BPH-addressing information). Because it is not obvious in another BPH domain for what kind of user the packet is intended for, an address-id is necessary. This address-id determines if it is a NAPs or a NAPe address.

The third and fourth field are added by the originating BPH. This information is needed when a packet has to be routed back from the destination user to the originating user in another BPH domain.

With this addressing scheme it is possible to address every user in the network in a unique way. This way of addressing is more independent of a possible implementation. The assignment of the addresses to BPHs and the distribution of the routing information will be a task of the configuration management (see section 9.5).

## 5.4 Topology

To meet the architectural requirements of the network service, a definition of a suitable topology is necessary. When defining the topology for the network of BPHs, the following problem has to be solved.

> Given the locations of the BPHs to be interconnected, and given the (expected) traffic between these BPHs. Find a suitable way of interconnecting these BPHs, in such a way that the functional requirements can be met with minimal costs.

It is assumed that the traffic is equally distributed over every BPH serving users. In section 5.1 the requirements, the network has to meet, are mentioned. Most of the architectural requirements have impact on the topology design. These requirements have to be realized, given the BPH as it is designed. The fact that this BPH is already present, puts the following constraints on the topology design.

- For reasons of performance there is a maximum number of BPHs, which may be present in the physical path between two users of the network. This number of BPHs is dependent on the application accessed by the user, and the delay caused by one BPH. It is defined that no more than three BPHs may be present in the physical path between two network users.

- A BPH has a maximum of 16 interfaces to interconnect. When such an interface is used, a DLC has to be offered (thus 15 sets). This physical limitation has great impact on the topology design.

- For the interconnecting of BPHs, permanently established lines will be used. In this way no extra CPU loading (apart from the network initialization) will be needed.

The design of the topology is split into three design-phases. In this way the topology design problem is decomposited into smaller problems which are easier to get grip on. The phases are :

1. Phase 1, Topology for network in one switch with one server
2. Phase 2, Topology for network in one switch with several servers
3. Phase 3, Topology for network in several switches with several servers

### 5.4.1 Phase 1

In this situation only one server is present (which may offer several applications) in one Sopho switch. The following definitions apply to this topology.

1. A primary BPH is a BPH to which the server is attached. BPHs and sets may also be attached to the primary. There is only one primary (because there is only one server).
2. A secondary is a BPH attached to a primary. Only BPHs or sets may be attached to these.
3. A ternary is a BPH attached to a secondary. Only sets may be attached to these.

Because the requirements stated in section 5.1 should be taken into account, and only one server is present, a centralized star topology seems a very suitable solution (see figure 5.7). In figure 5.7 the configuration is depicted which serves a maximum number of sets. In this case only to the ternary BPH, the sets are connected. The other BPHs interface with BPHs. In this way a maximum of

$$225 * 16 * 16 = 57600\ sets^2$$

[2] 225 Sets connected to a ternary BPH. 16 of these BPHs connect to a secondary BPH. 16 secondaries connect to a primary

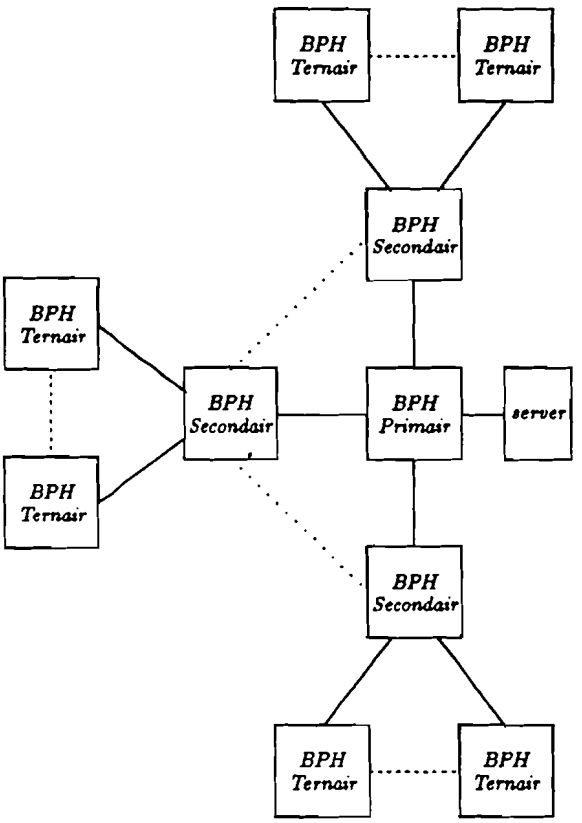



Figure 5.7: *Topology with one server*

can be served. This maximum will never be needed in a practical situation, and therefore not all the ternary BPHs will be needed (sets can also be attached to secondary BPHs).

This topology assures a maximum of 3 BPHs in the path from set to server. It also delivers a fair distribution of delays and traffic, due to its symmetrical structure. The routing of packets to the server can be kept very simple. This is only concerned with the forwarding of the packets on the outgoing link. This topology is not robust, in that it is very sensitive for BPH or link failure. No alternative routing can be applied in case of such a failure.

When more servers need to be added, this topology also shows some drawbacks. To meet the requirement of having only 3 BPHs in the path between set and server, the server may only be added at the primary BPH. This means that no more than four servers can be supported, or a longer delay must be accepted when adding the server to a secondary

or ternary. In the next section a topology which evolves from this topology will be defined. It does not have the drawbacks mentioned in this section.

### 5.4.2 Phase 2

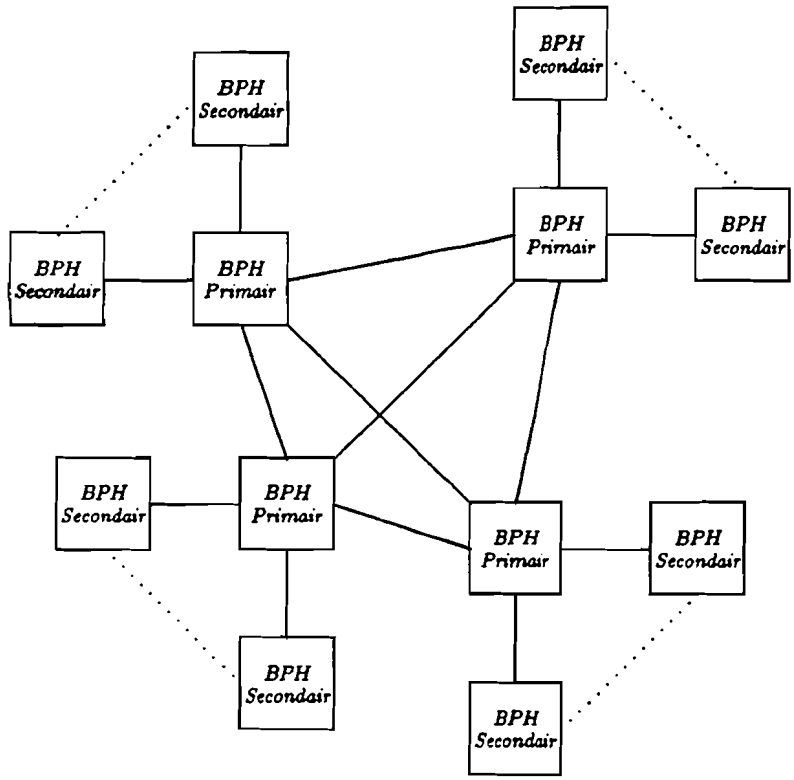



Figure 5.8: *Topology with several servers*

The following definitions apply to the topology which is derived in this section.

1. There are *primary* and *secondary* BPHs
2. A primary BPH is a BPH to which a server or a BPH is connected
3. A secondary BPH is a BPH which is connected to a primary BPH. Only DLCs are connected to this BPH
4. A secondary BPH becomes a primary BPH when a server or a BPH is connected to it

5. Primary BPHs are fully interconnected. This takes care of a maximum path-delay of 3 BPHs

Note that it is still possible that sets are attached to a primary BPH with this set of definitions. In figure 5.8 this topology is illustrated. A server can be connected to any primary BPH in the network.

More links are used to interconnect primaries, this means that connections for sets have to be sacrificed. Using this strategy, the maximum number of sets which can be served with this topology can be calculated as follows. Suppose there are $P$ primaries. This means that they have $P-1$ links to interconnect (full-mesh). There are now $(16-(P-1))$ links left to connect secondaries. The number of sets is :

$$Number\ of\ sets\ =\ P*(16-(P-1))*15*15 = 3825P - 225P^2$$

$$(15\ sets\ per\ DLC,\ 15\ DLC\ per\ BPH)$$

The maximum is found for the value of P, where the derative is zero :

$$3825 - 450P = 0 \ \Rightarrow\ P = 8.5$$

Thus for 8 or 9 primaries the maximum number of sets is 16200. In figure 5.9 the number of sets which can be served related to the number of primaries in the network is shown.

The maximum number of sets which can be served (16200) is much less than the number which was derived in the previous section. This number is however large enough for practical implementations. The topology offers a fair distribution of delays and traffic, because of its symmetrical structure. It is robust in that it allows alternative routing (shadow routing tables), in case of link or BPH failures. Thus by offering more robustness, flexibility and reliability, less sets can be served. The topology allows a flexible growing of the network, by applying the definitions given in the beginning of this section. It is best to determine first how many sets should be able to access a certain application. This number of sets can be translated into the number of primaries needed, via figure 5.9. In this way it is known how many interfaces should be reserved for the fully interconnection of primary BPHs. These may not be used to add secondaries, otherwise the network cannot grow to its maximum size.

### 5.4.3 Phase 3

When the network is distributed over several Sopho switches, a new demand restricts the freedom of the design of the topology. The number

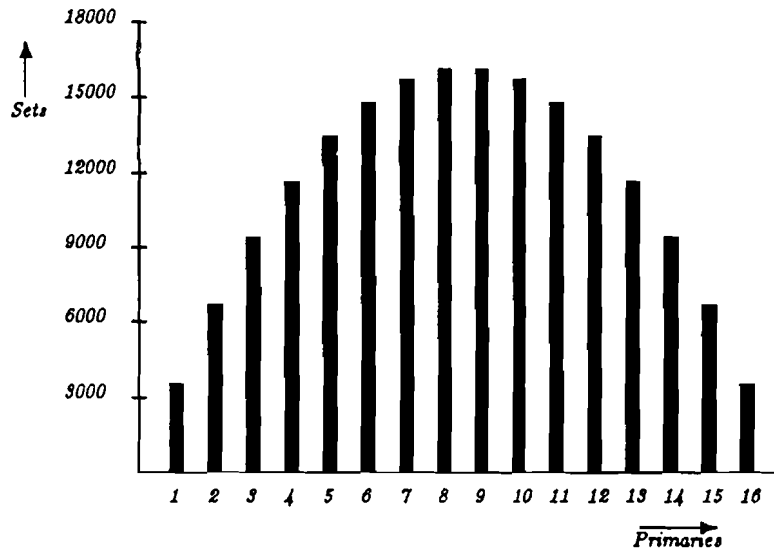



Figure 5.9: *Number of sets related to number of primaries*

of links between the BPHs in different switches should be kept as low as possible, due to the costs related to these links. The best possible solution is that only one link is used for the interconnection of BPHs in different switches.

When no more than four servers are present in the network, the topology as was derived for phase 1 suits well. Every BPH may be present in a different switch, and thus it is possible to form a network of $256^3$ switches. This number is only valid when no more than the maximum number of sets (225) are served by a BPH. In this case a network of BPHs should be present in the switch, to be able to allow more sets to access the servers. This can be achieved when a secondary BPH, together with its ternary BPHs, are present in one switch (see figure 5.7). This allows a maximum of 3600 sets to access the servers. This number is large enough for present switches.

When more than four servers are used, the topology derived for phase 2 is applicable. The primary BPHs should be present in one switch, for these need a lot of links to interconnect. The secondary BPHs can be placed in different Sopho switches. In this way a maximum of $72^4$ switches can be served. A problem arises when more than 225 sets need

[3] 16 secondary BPHs multiplied by 16 ternary BPHs

[4] maximum is obtained for 8 primaries and 9 secondaries

to be served by a BPH in a switch. The choice should the be made to make this secondary a primary (and thus more links between the switches) or allowing a path delay of maximal 4 BPHs to a server. The distribution of servers can than be chosen so that the application which is not time critical has the longest path.

### 5.4.4 Conclusions

It is obvious that there is no best solution. Depending on the situation and the priorities a certain topology will serve best. If a topology is needed which is allows alternative routing, and thus more reliability the solution for phase 2 will do best. The drawbacks are the usage of more links and thus more costs. If minimal links will be used, solution one is a good alternative. Special attention should be given to the adding of servers. It is best to keep this number as low as possible, because this implies minimal costs. The possibility to integrate different applications into one server should be investigated because this can greatly reduce the costs for networking.

# Chapter 6

# The Protocol for Interconnection

## 6.1 Introduction

One aspect which have not been treated yet, is the interconnection protocol to be used between two communicating BPH peers. A reliable means to transfer datapackets between communicating BPHs, is necessary to offer a means of communication which can meet the functional requirements of the network service. As it was chosen not to use a connection oriented way of datatransfer (see section 6.2), the Network Access Protocol for interconnection (NAPi) of BPHs can be kept very simple (only routing is performed). The underlying layer, the Link Access Protocol for interconnection (LAPi), offers a reliable, sequenced, flow-controlled link to NAPi. In every BPH there is a full layer 2 termination. At layer 3 the addressing information is examined and the packet is forwarded to the right outgoing link. Apart from the very thin Napi intervention, the service which is offered by the protocol for interconnection can thus be seen as a proprietary *frame switching service* (see section 4.1.7).

The Physical Access Protocol for interconnection (PAPi) has to be taken as it is, because the board has already been designed. The interface between the three mentioned layers is specified by defining the primitives which are used for layer-to-layer communication.

A conceptual model of the B-channel Packet Handler showing the different protocol stacks, is depicted in figure 6.1[1]. Each block in this

[1]The interface with management is omitted in this figure, this will be dealt with in chapter 9.

model represents an entity. This entity can either be a communication entity, or an interworking entity. The internal routing entity is respon-

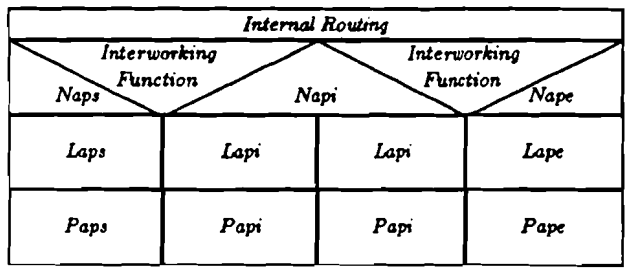



Figure 6.1: *The model of the BPH*

sible for the correct transfer of datapackets between the different layer 3 entities. It needs to be able to address each entity individually. This depends on how the different entities are going to be implemented, and will therefore be no part of this study.

The layer 3 entities exchange packets with the intervention of the interworking function entity. The service offered by this entity takes care of the correct translation of the different formats. In section 6.6 this interworking function will be treated. An informal specification of this function will also be given there. Because LAPi is the most complicated layer it is formally specified. With this specification it is possible to simulate the protocol definition, in order to check the correct working. This will be dealt with in section 6.5.

## 6.2 The Network Access Protocol for Interconnection

As it was chosen not to use a connection oriented way of data transfer between the BPHs, there is no need for Napi to establish a connection with its peer entity. NAPi can thus be kept very simple, it only has two states (see figure 6.2). Napi can only perform transitions between these states by means of management intervention. Management is responsible for putting everything into service. Layer 3 should be taken into service, after layer 1 and layer 2 have established communication with their peer entities. When layer 3 is not working properly it is up to management to initiate the right action. Because the protocol for interconnection offers a different link for transferring datapackets

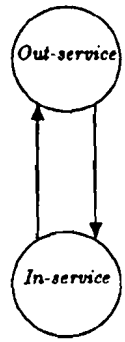



Figure 6.2: *State machine of a NAPi entity*

from users and management, there are two NAPi entities present in each BPH. These entities are accessed by LAPi by means of a different Service Access Point Identifier. The management-NAPi entity can receive datapackets from the Sopho switch , which should be transferred to a different switch. The internal routing entity is responsible for the routing of these packets to the right NAPi entity. There is no need for having a different NAPi entity for each link, because no association is present between two peer layer 3 entities.

The only activity which is performed in layer 3 is the routing of the datapackets. NAPi can either receive an *indication* from a LAPi entity, or a *request* from the interworking function, to transfer a datapacket. The used Napi packet is of the format as shown in figure 6.3. The

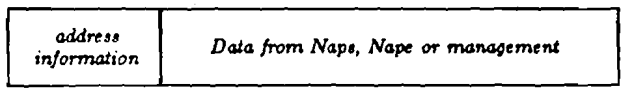



Figure 6.3: *The NAPi datapacket format*

address information is as defined in section 6.3. According to the destination address, the packet is routed to the right outgoing link. For the routing of the packets, a routing table which holds the relations between the BPH-destination address and the right outgoing link, is necessary. This table should be present in every BPH. Management must be able to perform actions on this table in order to update the routing table when configuration changes have taken place. As a first implementation a fixed routing table can be used. Shadow tables for applying alternative routing, can make the network service more reliable. The routing algorithm to be used is beyond the scope of this report, and is for further study.

When the datapacket is at its destination, it is passed to the interworking function. This interworking function removes the address information and passes the remaining layer 3 data, along with the address of the right entity to the internal routing entity. This entity relieves Napi from routing in the BPH-domain.

Because there is no different Napi entity for each link, the Napi entity does not maintain an association with its peer entity. Napi just forwards the packet to the right LAPi entity, without expecting any confirmation. This works well as long as LAPi is always able to receive a packet from NAPi. This demand can be met with proper memory management, and should not cause any problem. Because Napi offers a very primitive service, Lapi should offer a more sophisticated service to Lapi, in order to achieve a reliable interconnection service.

NAPi communicates with a LAPi entity by means of the exchange of the following primitives.

- *dl-data-req* : used by Napi to transfer datapackets to Lapi
- *dl-data-ind* : used by Lapi to deliver a datapacket received to Napi.

Along with this primitives the datapacket as depicted in figure 6.3 is passed. Because there is no association between two peer layer 3 entities, there is no need for a special peer-to-peer protocol. Therefore no protocol messages are defined for NAPi.

## 6.3 The Link Access Protocol for Interconnection

In this section a description is given of the layer 2 protocol which will be used to transfer the layer 3 datapackets. Lapi will accept a datapacket from Napi and will try to deliver it to the peer BPH. No acknowledgement is given by Lapi to Napi when a datapacket is correctly transmitted.

The service provide by Lapi to Napi can be described as follows.

- Provision of 2 datalink connections for acknowledged transfer of Napi datapackets (either data or management).
- Identification of datalink connection endpoints. Because layer 3 has a routing function it needs to identify the different datalinks in order to choose the right entity for the right outgoing link.

- Sequence integrity.
- Flow control via sliding-window mechanism. The sender has a maximum, fixed windowsize of k. The receiver uses a windowsize of 1.
- Notification to management in case of unrecoverable errors.
- Notification to peer entity of some errors, like loss of sequence.

In order to provide this service to Napi, the Lapi protocol is defined. This protocol allows a more efficient use of the interconnection links, by

1. allowing several data links over a physical link. This is achieved by layer 2 multiplexing,
2. using a windowing mechanism. This enlarges the throughput of the link.

LAPi uses a sliding-windowing mechanism with windowsize k. This window size is fixed, during the operation of the layer, but it may be changed by management. A windowsize of k frames means that there can be outstanding k I-frames which have not yet received an acknowledgement from the peer. This mechanism assures a higher throughput because there is no need to wait for an acknowledgement when the window is not yet fully occupied (see figure 6.4).

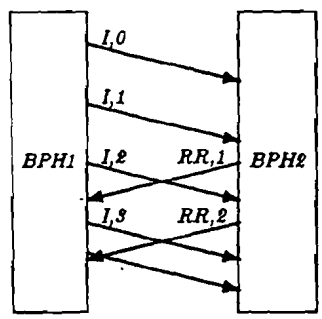



Figure 6.4: *Example of use of the sliding window protocol*

The choice of the windowsize is very important because when it is chosen too small the performance degrades, and when it is chosen too large, too much resources are reserved. For the interconnection of BPHs a digital link of 64 kbit/s is used. Suppose the BPH is capable of handling N frames per second, and the frame length is L bits. When BPH1

sends a frame to BPH2, the frame takes $\frac{L}{64.10^3}$ to arrive at BPH2, and BPH2 takes $\frac{1}{N}$ seconds to handle the frame and send an acknowledgement (see figure 6.4). BPH1 which receives the acknowledgement also takes $\frac{1}{N}$ second to handle the frame. So when a windowsize of one is chooses every

$$\frac{2}{N} + \frac{L_I + L_{ack}}{64.10^3} \; second,$$

a new frame can be sent. The second factor in this delay can be neglected because it is much smaller than the delay caused by the BPH. When a windowing mechanism is used however, every $\frac{L}{64.10^3}$ second a frame can be sent (the BPH continuously puts frames on the link). The windowsize k can thus be determined as follows.

$$k = \frac{Delay\ caused\ by\ BPH1 + BPH2}{Time\ to\ put\ frame\ on\ link} \Rightarrow k = \frac{\frac{2}{N}}{\frac{L}{64.10^3}} = \frac{128.10^3}{N * L}$$

If the BPH is able to handle 25 frames per second, and the length of a packet is 100 bytes [ref..] then the windowsize should be more than 6.

The link access protocol for interconnection is decomposited into two sublayers. Layer 2A takes care of the lower layer 2 functions, like error checking, obtaining transparency etc. Layer 2B is the upper part of layer 2, and takes care of the correct handling of the LAPi peer-to-peer protocol (see figure 6.5). These layers will now be dealt with.

### 6.3.1 Upper part of LAPi, layer 2B

Layer 2B takes care of the correct functioning of the peer to peer protocol. Because layer 1 implies a bit oriented protocol, Lapi will is a HDLC[2] like protocol. In order to be able to provide a sequence controlled, reliable communication link to Napi, there is the need for state variables which take control the sequencing and the correct layer operation. For layer 2B the following variables are needed.

1. VS : This is the send state variable. VS denotes the sequence number of the next I-frame to be transmitted.
2. VA : This is the acknowledge state variable. VA identifies the sequence number of the last transmitted I-frame which was acknowledged by the peer. Frames with a sequence number less than VA can be discarded.

[2]High-level Data Link Connection protocol, see [60].

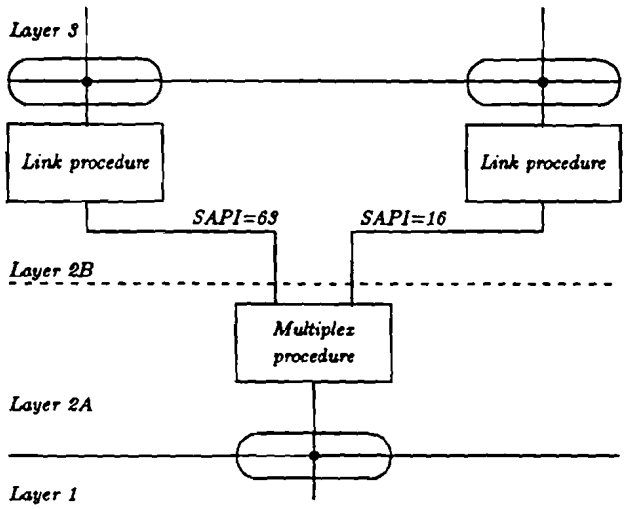



Figure 6.5: *Logical division of layer 2*

3. VR : This is the receive state variable. VR denotes the sequence number of the next I-frame expected to be received.

4. NS : NS is the send sequence number, and is contained by I-frames. The value of NS equals VS when an I-frame is transmitted.

5. NR : NR is the receive sequence number, and is carried by RR and RNR frames. This number denotes the sequence number of the next I-frame which is expected. It is used to acknowledge the receipt of I-frames.

As was stated before, layer 2B defines the peer-to-peer protocol for interconnection. This protocol uses existing agreements for the naming of the different protocol messages [69]. The following peer to peer messages are defined.

**SABM** The Set Asynchronous Balanced Mode command is used to reset the peer. The peer data link entity should acknowledge this command with an UA response. The state variables VR, VA and VS have to be set to zero on acceptance of this command. No information is carried with this command. I frames which are not acknowledged on receipt of an SABM command are discarded. Management should take care of the recovering of the lost I frames.

**UA** The Unnumbered Acknowledge response is used by the link entity to acknowledge the receipt and acceptance of an SABM command. The transmission of this response indicates that all state variables have been reset and that any busy condition is cleared. No information is carried with this response.

**I** The Information command is used to transfer sequenced NAPi packets to the communicating peer entity, across the data link. This command is not used to acknowledge other received I-frames.

**RR** The Receiver Ready command/response is used to

- acknowledge the receipt of one or more I-frames (up to NR-1).
- indicate the clearing of a busy condition
- indicate that the entity is able to receive more I frames

**RNR** The Receiver Not Ready command/response is used to indicate a busy condition. Until the clearance of this busy condition, no more I frames may be transmitted by the peer. It also acknowledges I frames up to NR-1.

Layer 2B assures the sequence integrity by checking the sequence numbers of the frames received, with the value of the state variables. Depending on the result of this check, normal operation can proceed, or an error condition is exists.

The error correction is established by means of the *Go Back N* algorithm [3]. Use is made of a timer, which is set to a value large enough, to allow the acknowledgement, to arrive before the timer times out[3]. If a frame is not correctly received, layer 2B will never get an indication. This means that the timer will expire. In this case the first frame which is not acknowledged (denoted by VA) is retransmitted. The peer will eventually acknowledge this frame. The sequence number carried in this frame (NR) determines, which I-frames also need to be retransmitted (frames with sequence number in range {NR..VS}). The following is needed to allow layer 2B to perform the error correction described.

1. T200 : T200 is the timer which is set when a frame is transmitted to the peer. The value to which this timer is set, together

[3]This value should therefore be larger than $\frac{2}{N} + \frac{L_I + L_{ack}}{64.10^3}$ seconds. See section 6.3

with N200, determines the time before an error is signalled by management.

2. N200 : N200 denotes the maximum number of retransmissions which may take place before management is informed about the occurrence of an error.

3. rc : The retransmission counter (rc), denotes the number of times a frame is retransmitted. The value of rc should not exceed the threshold (N200). If it does, the management will be informed.

Layer 2B implements a lot of the service, which Lapi offers to Napi. To structure the design of layer 2B it is decomposited into several states.

**out-service** : In this state layer 2B is waiting for management, to initiate the parameters and take the layer into service.

**in-service** : In this state layer 2B has set its parameters, and is waiting for the command to synchronize with its peer.

**wait-estab** : An SABM-frame has been sent by the layer 2B entity. In this state the UA response should be received.

**info-transf** : The two communicating entities are synchronized, and information transfer can take place.

**recover** : There has been a timeout. The frame is retransmitted, in order to go back to normal communication.

The state machine of layer 2B is as shown in figure 6.6.
In figure 6.7 it is show on which primitives the state transitions as shown in figure 6.6 take place.

Layer 2B communicates with layer 3 and layer 2A. The primitives between layer 3 and layer 2B are given in the previous section. Because this interface is not between two different OSI layers, proprietary naming for the primitives is used. The following primitives are defined between layer 2B and 2A.

- *frame-received*
- *transmit-frame*

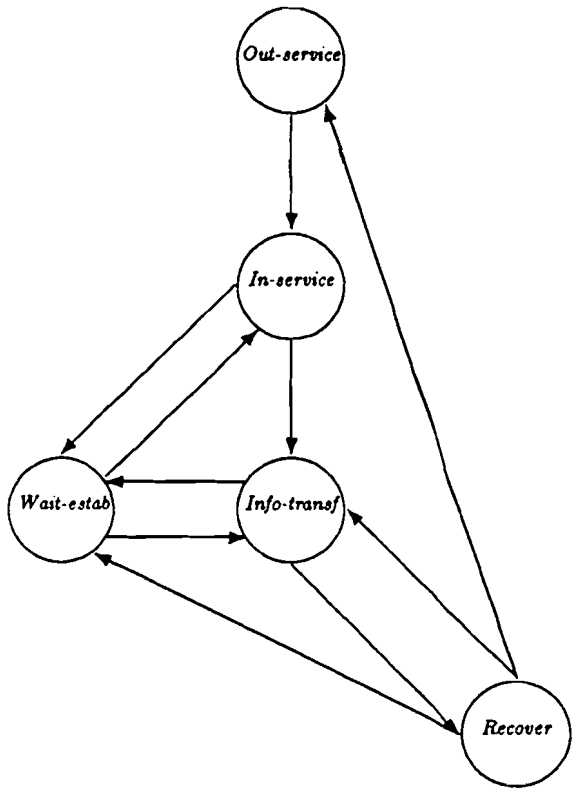



Figure 6.6: *State machine of layer 2B*

Along with these primitives the frames are passed. All the layers interface with management. Layer 2B has a rather advanced interface with management. The establishing and releasing of a data link connection are tasks from management, because layer 3 does not offer this functionality.

A formal description of layer 2B, which specifies the working of this layer is discussed in section 6.5.

### 6.3.2 Layer 2A

This part of layer 2 is concerned with the lower layer 2 functions. It performs a multiplexing function, so that the link for interconnection is more efficiently used. The distinction between the different datalinks is made by means of a Service Access Point Identifier. Two different SAPIs are defined.

| State | Primitive | Condition | Nextstate |
|---|---|---|---|
| Out-service | take-in-serv | | In-service |
| Out-service | * any other * | | Out-service |
| In-service | dl-est-req | | Wait-estab |
| In-service | frame-received | SABM | Info-transf |
| In-service | * any other * | | In-service |
| Wait-estab | frame-received | UA | Info-transf |
| Wait-estab | T200 | rc $\leq$ N200 | Wait-estab |
| Wait-estab | T200 | rc > N200 | In-service |
| Wait-estab | * any other * | | Wait-estab |
| Info-transf | dl-est-req | | Wait-estab |
| Info-transf | mdl-contr-req | | Info-transf |
| Info-transf | dl-data-req | | Info-transf |
| Info-transf | frame-received | RR,RNR VS<NR< VA | Wait-estab |
| Info-transf | frame-received | *any other * | Info-transf |
| Info-transf | T200 | | Recover |
| Recover | dl-est-req | | Wait-estab |
| Recover | mdl-contr-req | | Recover |
| Recover | dl-data-req | | Recover |
| Recover | frame-received | RR,RNR VS<NR< VA | Wait-estab |
| Recover | frame-received | RR,RNR VA $\leq$ NR $\leq$ VS | Info-transf |
| Recover | frame-received | I, SABM, UA | Recover |
| Recover | T200 | rc $\leq$ N200 | Recover |
| Recover | T200 | rc > N200 | Out-service |

Figure 6.7: *The state transitions of layer 2B*

1. SAPI 16 is used for the transfer of data destined for users of the network service.

2. SAPI 63 is used for the transfer of management information between different Sopho switches

Another advantage of using different SAPIs is that the transfer of time critical management information is not affected by the datatraffic. Of course other SAPIs may be used if the need arises to support more data links for information transfer.

Layer 2A will also check on transmission errors. This is done by calculating a Frame Check Sequence, and adding this to the frame. The FCS is calculated by means of a Cyclic Redundancy Check [3]. The polynom to be used for this calculation is the CCITT polynom (X16+X12+X5+1). By bitstuffing transparency is obtained. No more than five successive logical ones may succeed each other in a frame. The chip which is used for communication between the BPHs (PCP, see next section), takes care of these lower layer 2 functions (bitstuffing and CRC calculation). It is up to layer 2A to check this CRC with a predefined value. If the FCS is correct it will pass the frame to the right link-entity in layer 2B, according to the Service Access Point Identifier.

The layer 2 frame format is as indicated in figure 6.8.

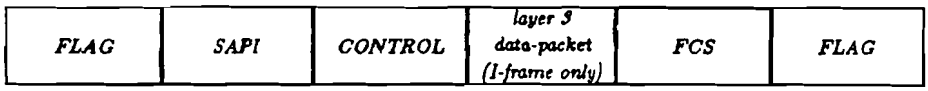



Figure 6.8: *Layer 2 frame format*

**Flag** : The flag denotes the beginning and end of a frame. By violating the coding rules, this flag can be identified. The PCP-chip used for communication between the BPHs determines that this flag is 01111110. Thus more than 5 successive logical ones are used to violate the coding.

**SAPI** : To differentiate between the datalinks present on the physical link.

**Control** : Because there are five different peer to peer messages the coding of these will only need three bits. This means that five bits can be used for sequencing. The control field could be coded as in figure 6.9.

**FCS** : The Frame Check Sequence is added to perform error detection

Layer 2A interfaces with layer 2B and layer 1. Primitives between layer 2A and 1 :

- *ph-data-req*
- *ph-data-ind*

## 6.4 The Physical Access Protocol for Interconnection

When interconnecting BPHs use is made of the hardware already present on the board. Instead of connecting a Digital Line Circuit to serve sets, a BPH can be connected to this interface. Thus by sacrifying sets, BPHs can be attached. The physical interface is realized by the *Protocol CoProcessor (PCP)* [73]. The PCP is a custom integrated circuit capable of handling sixteen communication channels that are multiplexed into

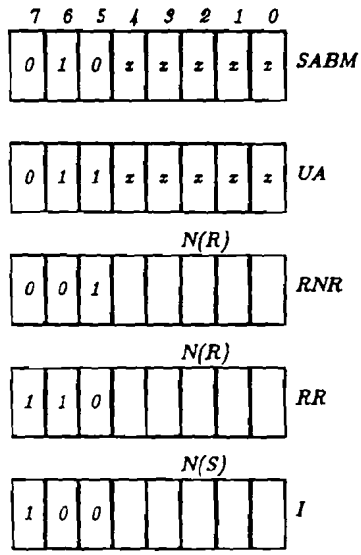



Figure 6.9: *Possible coding of the control field*

a 2 Mbit/s datastream. The communication channels are set to *BOP mode* in order to serve bit-oriented protocols, and are *full duplex*. The PCP takes care of

1. Flag transmission and flag detection

2. Zero bit deletion and insertion to obtain transparency

3. Cyclic Redundancy Check generation and computation according to the X16+X12+X5+1 polynom.

So also most of the lower layer 2 functions are performed by this chip.

The interface of the PCP and the host processor is such that minimal processor loading is required. When the PCP has received data it interrupts the processor and gives the address where the data can be found. For detailed information about this chip reference [73] should be studied.

## 6.5 Formal description of the protocol for interconnection

The description given of the protocol so far, merely describes the architecture of the protocol. Except from layer 1, which is already implemented, the description is too abstract to use for an implementation. Therefore a description of the protocol should be given which can meet the following requirements.

1. The protocol should be formally specified, and its correct working should be checked.

2. The specification must be usable as a guide for implementation.

3. If possible, use should be made of specification methods supported by Sopho S Low Range.

Two solutions applied well, to describe the protocol.

**Milner 's Calculus of Communication Systems** CCS is a formalism which can be used to specify communication protocols by means of algebraic expressions [59]. Due to the lack of tools for CCS, and its desultoriness, it was not considered a valid solution.

**CCITT Specification and Description Language** SDL has been designed by the CCITT as a language which can be used to specify and describe the behavior of telecommunication systems [62]. Due to the fact that

- It is possible to give a formal description of the protocol with SDL,
- the Graphical Representation of SDL is very easy to understand,
- SDL is supported with several tools by Philips (graphical tools, simulation tools)
- and a description in SDL is a good guide for implementation (there are tools for translating an SDL description into C)

this solution for specification was chosen.

The reader is referred to reference [62], for a description of the SDL language. Because of the complexity of Lapi, this layer is formally specified. With this formal specification, a simulator is generated, which is used to check the correct operation of this layer. It was not considered necessary to specify Napi, because of its very simple structure. In the next section it is depicted how the protocol was modelled in SDL. In appendix A, the complete graphical representation is given.

### 6.5.1 The protocol in SDL

In this section it is depicted how the protocol is modelled in SDL. The layered structure which is used for the protocol, is also used in the SDL description. In the model a layer is represented as a *block*. In SDL blocks are interconnected by means of *channels*. These channels carry *signals*, which are the actual means by which communication take place. The primitives used for communication between the layers, are signals in SDL. A block can contain one or several *processes*. These processes contain the actual *statemachine*, which describes the operation of the protocol. The structure of the layer 2 in SDL, is shown in figure 6.10. In the complete SDL description, also layer 3 is represented as a block,

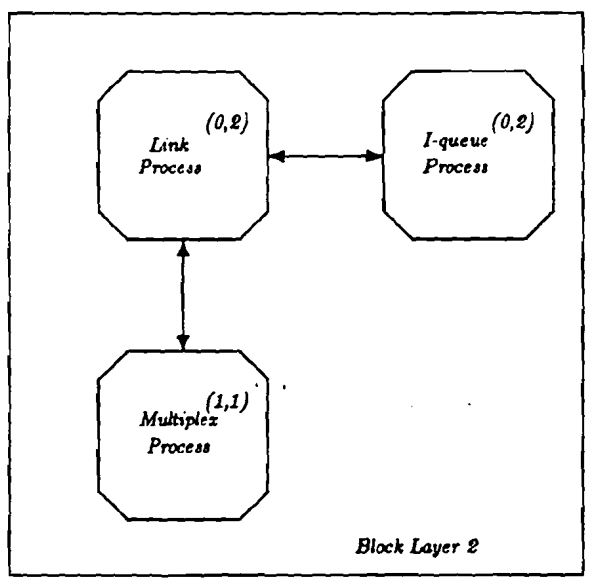



Figure 6.10: *SDL structure of layer 2*

containing processes. These processes are processes which only pass

information to the right entity. They are needed for simulating purposes. In figure 6.10, it is shown that layer 2 contains 3 processes. The processes *Link* and *Multiplexing*, perform the function as described by layer 2B and layer 2A respectively. Every process owns information about the number of instances which is present on system initialization, and about the maximum number of instances which may exist simultaneously. This is denoted by the numbers in the topright of the process symbol. For instance (1,1) means one process exists at most, and it is created on system initialization. Two link processes may exist (one for data and one for management). These processes are created by a special process (not shown in the figure), which takes care of the correct administration for communication between processes[4]. The process I-queue is created by the link process, and it is an abstraction of the buffer to store I-frames.

One step lower in the hierarchy, it is seen what is inside a process. A process contains a statemachine. For the link process, the statemachine is shown in figure 6.6.

In SDL a transition to a new state (which may be the same state), is always performed on receipt of a signal. SDL also allows parameters to be passed along with these signals. An example of a statemachine in SDL can be found in figure 6.11. The specification of the protocol in SDL is found in appendix A.

### 6.5.2 The simulation

A special tool allows the generation of a simulator from an SDL specification. For this purpose the SDL description is translated into Pascal, which then need to be compiled. This is only possible however, if a formal specification is given. Because a formal specification of layer 2 of the protocol is made, it is possible to generate a simulator of this specification. An example of the results of the simulation of the establishing of a datalink is shown in figure 6.12. The simulator is a very powerful tool which can be used to verify the correct working of the specification. It is possible to write monitor programs which can act as input for the simulator. In this way it is very easy to generate exceptional conditions. The simulator also allows the setting of breakpoints, tracing signals, examining queues with signals for processes, etc. A check is performed if an output of a signal does result into a transition by a receiver. When no

[4]For example, the multiplexing process should know which link process is for management and which process for data.

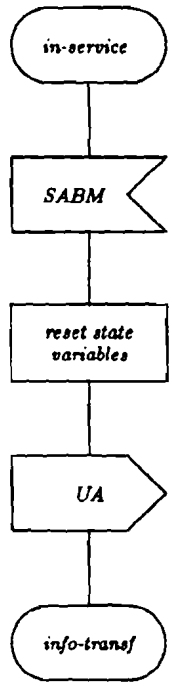



Figure 6.11: *Example of statemachine in SDL*

receiver processes are found, or the receiving process does not perform any action on the signal, an error is reported. So a very powerful tool is available for the checking of the correct operation of the description.

With the simulator the correct working of the layer 2 protocol has been verified. The simulations show that the description of Lapi in SDL, is able to deliver the service to Napi as described in section 5.3. In appendix B a simulator output from the protocol operation is shown.

## 6.6 The Interworking Function

As can be seen in figure 6.1, there is the need for an interworking function entity which offers a service to layer 3 entities. This service takes care of the correct translation of the different layer 3 formats. In this section this service will be described by means of an informal SDL description. The total description can be found in appendix C

In this section the different functions which need to be performed to deliver the service, will be described. This is achieved by means of

```
*** TRANSITION START
*      Pid     : Link_process:1
*      State   : in_service
*      Input   : frame_received
*      Sender  : multiplexing:1
*      Now     : 0.00
*      Parameter(s) : (. SABM .)
*   DECISION
*   TASK
*   OUTPUT of transmit_frame to multiplexing:1
*      Parameter(s) : (. UA .)
*   PROCEDURE START : res_state_var
*   TASK
*   TASK
*   PROCEDURE RETURN : res_state_var
*   OUTPUT of dl_est_ind to management
*
*** NEXTSTATE info_transf

*** TRANSITION START
*      Pid     : multiplexing:1
*      State   : IDLE
*      Input   : transmit_frame
*      Sender  : Link_process:1
*      Now     : 0.00
*      Parameter(s) : (. UA .)
*   DECISION
*   TASK
*   TASK
*   OUTPUT of ph_data_req to layer1
*      Parameter(s) : (. frame .)
*** NEXTSTATE in-service
```

Figure 6.12: *Example of a simulator output*

a higher level description in SDL. The reader is assumed to be able to understand the SDL graphical syntax ([62]).

The interworking function is designed in such a way, that its service offered, allows easy adaption to new protocol stacks defined. In the following only the interworking functions for the protocol stacks as already defined, will be treated.

The interworking function needs the following information, in order to work properly. This information should be delivered by management on network initialization. It should also be accessible by management, because it may change during the operation of the BPH.

**PVC-table :** This table holds the relation between the linkaddresses of the set, and the logical channel number to be used on the link between BPH and server. The range of the logical channel number is between 1 and 450, because no more than 450 ports can attach to a BPH, when this BPH is also connected to the network. Logical

channel number 0 is used for management of the server.

**Application-table** : This table holds the relation between the logical channel numbers of the applications, and the BPH addresses to which the server, which supports the application, is attached.

**Header-table** : This table is used for the tempory storing of the address header in relation to the logical channel number which is used for the BPH server link. When a packet arrives at a BPH (via the network) to which a server is attached, the address header need to be stored, in order to be able to route the packet which comes from the server, back to the originating user. Because Nape uses a logical channel number over the link between server and BPH, a lcn should be assigned for these packets. The range of these lcns is between 451 and 4095. The maximum of the lcn value is determined by the address space of Nape packets. The minimum value is imposed by the fact that users connecting directly to a BPH uses the range from 1 to 450. In this way the interworking function can decide if it should sent the packet to a Naps, or Napi entity.

**Own-BPH address** : Every BPH should know its own address. This address should be added in the address header of a Napi packet. It is needed to route the packet to the user which initiated the application access. This address is also needed by Napi, to conclude that the packet has arrived at the destination.

### 6.6.1 Interworking on receipt of Naps data

When, after examination of the logical channel number (lcn) of the application accessed, it is concluded that the application is not served by a server in the BPH domain, the need arises to use the network service. Before the data can be given to a Napi entity however, a header which contains all the necessary addressing information (as shown in section 6.3), need to be assembled. The total packet, which consists of the addressing information, and the Naps layer 3 data, will then be given to the local routing entity, which selects the right Napi entity (see figure 6.13).

When the server with the application, is in the BPH domain, there is no need to use the network service. The only task which need to be performed is the translation of the link address of the set, to the lcn to

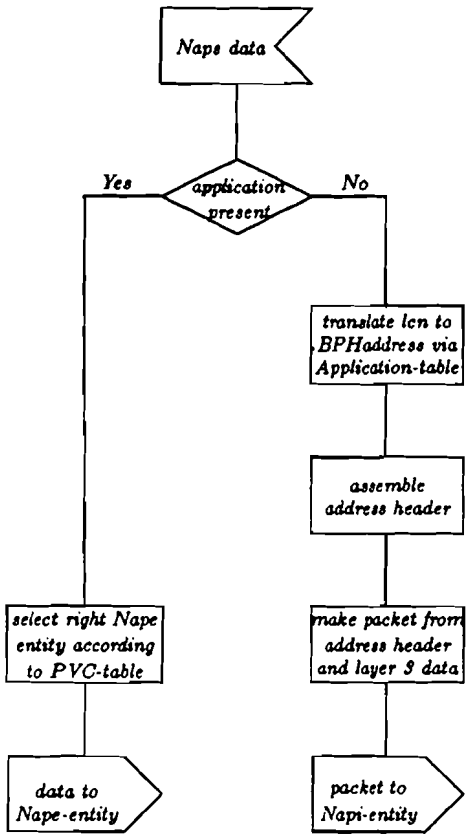



Figure 6.13: *Tasks on receipt of Naps data*

be used for the link between the BPH and server. Now the data can be given to the local routing entity, which selects the right Nape entity.

### 6.6.2 Interworking on receipt of a Napi packet

When a packet is received from a Napi entity, it is either destined for Naps or Nape. The destination field which describes the addressing information for the BPH domain is accessed, in order to acquire the needed routing information. The tasks needed to perform are depicted in figure 6.15.

### 6.6.3 Interworking on receipt of Nape data

When the lcn of the packet is in the range from 450 to 4095, the packet must be given to the Napi entity. The lcn can be used to retrieve

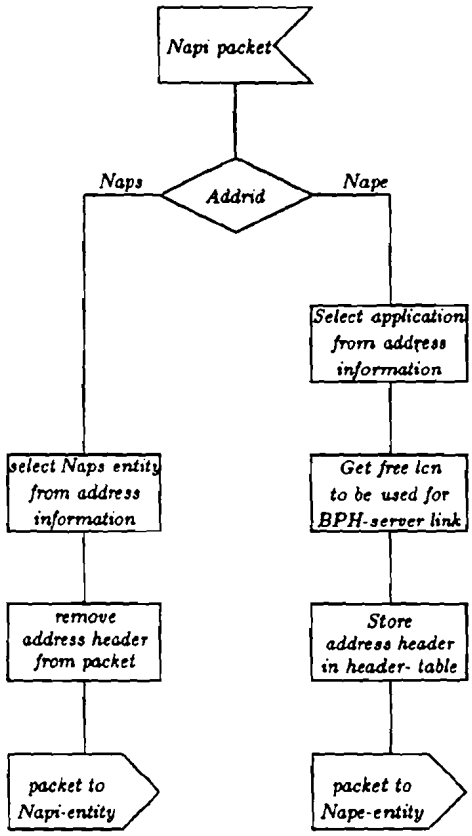



Figure 6.14: *Tasks on receipt of a Napi packet*

the header which was recently stored in the Header-table. The packet should be routed back to the originating user. Therefore the source and destination fields have to be swapped, because the old source becomes the new destination. The packet with the new header can now be given to the Napi entity (see Fig.6.16). When the packet is destined for the Naps entity, an access to the PVC table will give the right address.

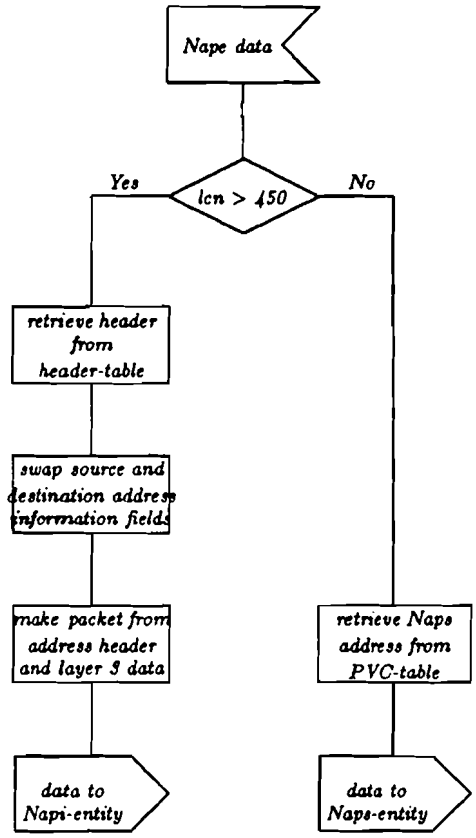



Figure 6.15: *Tasks on receipt of Nape data*

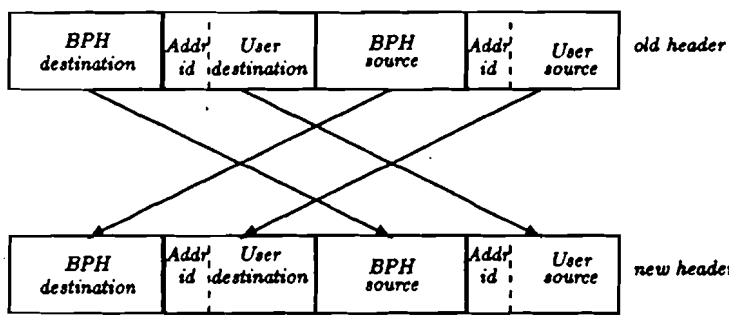



Figure 6.16: *The manipulation of the header*

# Chapter 7

# Management of the BPH network

## 7.1 Introduction

In this chapter the management of the BPH network is discussed. Due to the growing importance of networks (LAN, ISDN), network management is a very hot item in the standardization institutions. Network management for the network of BPHs is also very important. It is needed to provide a set of facilities, so that actions can be taken in reaction to certain events in the network, or that actions can be initiated by the management system in order to create a defined situation in the network.

Not only should be defined *what* is important to be managed (functional requirements), but also *how* a certain action initiated by management, can be performed on a resource (architectural requirements). To be able to perform management on the BPH network, the BPH should deliver an interface to the Sopho switch. The model of the BPH, which takes into account this interface is shown in figure 7.1. The management entity acts as the peer entity for the process, which controls the correct operating of the network. This entity performs all the functions which will be needed to meet the functional requirements of network management. The Layer Management Entity (LME) can be seen as the interface between the different protocol entities and management. The protocol stack towards Sopho S (Napm, Lapm, Papm) is proprietary, and is not dealt with in this report. For certain management activities, it can be necessary to obtain information from BPHs belonging to another Sopho S. This information can be transported via the management

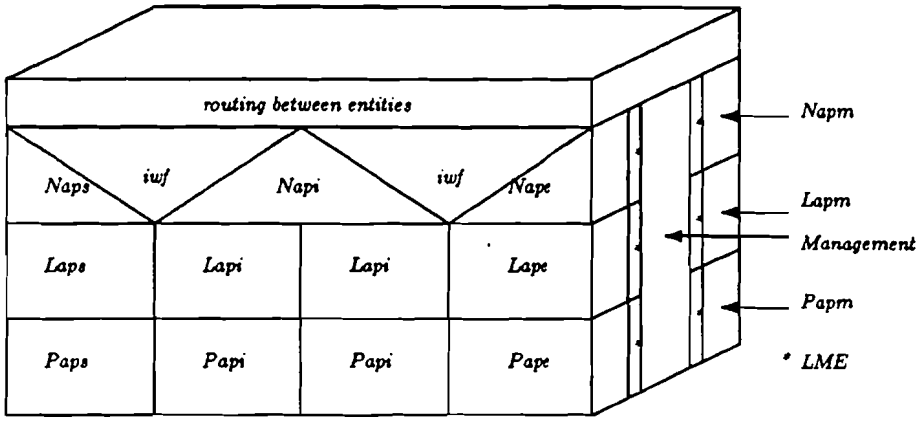



Figure 7.1: *The model of the BPH*

link which is offered by the protocol for interconnection. This infrastructure offered by the BPH can however also be used for the transfer of other management related information.

The management system do not necessarily have to be inside the Sopho switch. It may be a special terminal connected to the Sopho S. Every SOPHO S has its own management terminal connected to it. One terminal will act as the central management terminal, and contain the different processes. This management infrastructure will be dealt with in section 7.3.

In the next section the several requirements which the management must satisfy, will be treated.

## 7.2 Functional requirements

When applying network management there are a lot of task which should be done. The network of BPHs must be capable of being initialized, carrying datapackets, recovering from errors, reconfiguring in case of adding and removing of BPHs, etc. Also routing information should be present at each BPH, in order to allow a packet to arrive at its destination. This routing information should be updated when the network is configured. All these actions should be performed by management. To get a better view of the different aspects of network management, a functional decomposition of this problem is made. According to the principles used

by the standardisation institutes (OSI/ISO, ECMA, CCITT), the functional requirements for BPH network management will be treated. OSI recognized the following functional requirements [99].

**Configuration management** : Configuration management deals with the definition, the collection, the use and the correct management of the configuration data. Configuration management is responsible for the correct initialization of the network, but also for management actions which need information about the configuration. Configuration management provides facilities which allow the remotely initiating, resetting and closing down of total communication systems (BPH), but also entities belonging to such a communication system. With configuration management also means should be provided which allow the updating of routing information. Configuration management is a very important facility, which should certainly be present when management is defined for the network of BPHs.

**Fault management** : Fault management is the detection, diagnosis and correction of faults that may result in the malfunctioning of a part of the network, or even worse, the total network. It deals with short term solutions for errors occurring in the network. Errors should be notified to fault management. Therefore it is important that the different resources present in the network, provide an interface to fault management.

**Performance management** : Performance management deals with the long term operation of the network. Statistical data of the traffic distribution in the network should result in an architecture of the network, which delivers the best possible performance at the lowest costs.

**Security management** : Security management prevents data from being accessed by persons who are not classified to do so. Also the prevention of the deletion or changing of certain resources should be supplied.

**Accounting management** : Accounting management deals with the collection of billing data. For the BPH network this is not necessary yet, because it will be delivered as a service which can be accessed by the set freely.

The five different management domains mentioned above are all *system management* functions. They apply to the functioning of the system as a whole. These functions can work properly only if the different protocols, supply information to the different management functions. Therefore also *Layer management* functions are required. These functions take care of layer dependent items which are relevant for management. Finally there is *protocol management*, which deals with the management of two communicating entities.

The functional requirements are requirements as seen from an operator point of view, who wants to manage the network. For correct functioning of management a means should be provided to be able to perform the tasks. These will be dealt in the next section about the architectural requirements.

## 7.3 Architectural requirements

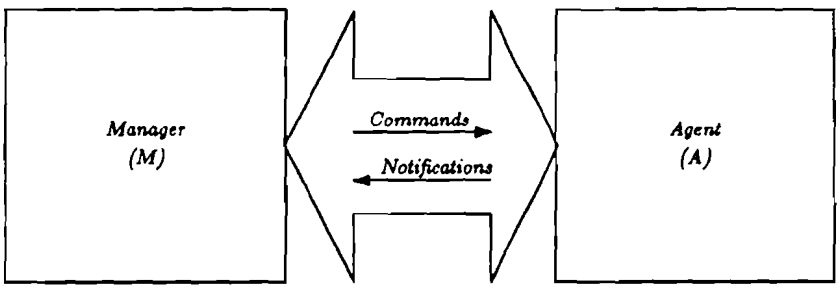



Figure 7.2: *The manager-agent relation*

The network of B-channel packet handlers may become rather large. For management purposes it is important, to define a structure in the network. In this way a better view can be obtained of the relations present in the network between the different resources. A good way of defining a structure in the network is to define domains. A domain is a logical decomposition of the total network. It can contain resources which have the same properties or functionality.

When applying management activities there exist a hierarchical relation between the originator of the action and the receiver. In the context of management the terminology *manager* and *agent* will be used.

> A manager is responsible for manipulating and monitoring the domains which it supervises.

> An agent performs management functions upon receipt of a command from a manager. Agents may also forward notifications to a manager, to report on the occurrence of certain events in its domain (see figure 7.2).

An iterative definition of manager and agents is used, this means that an agent can become a manager again for its domain. An agent has the capability to apply filtering to the events which occur in its domain. In this way not all events occurred are passed to the manager of the agent (or only after a certain amount of events). This can greatly reduce the loading of the manager. Finally a manager can have several agents, and an agent can also have several managers. It is up to the agents to decide which managers have to be informed about the occurred events.

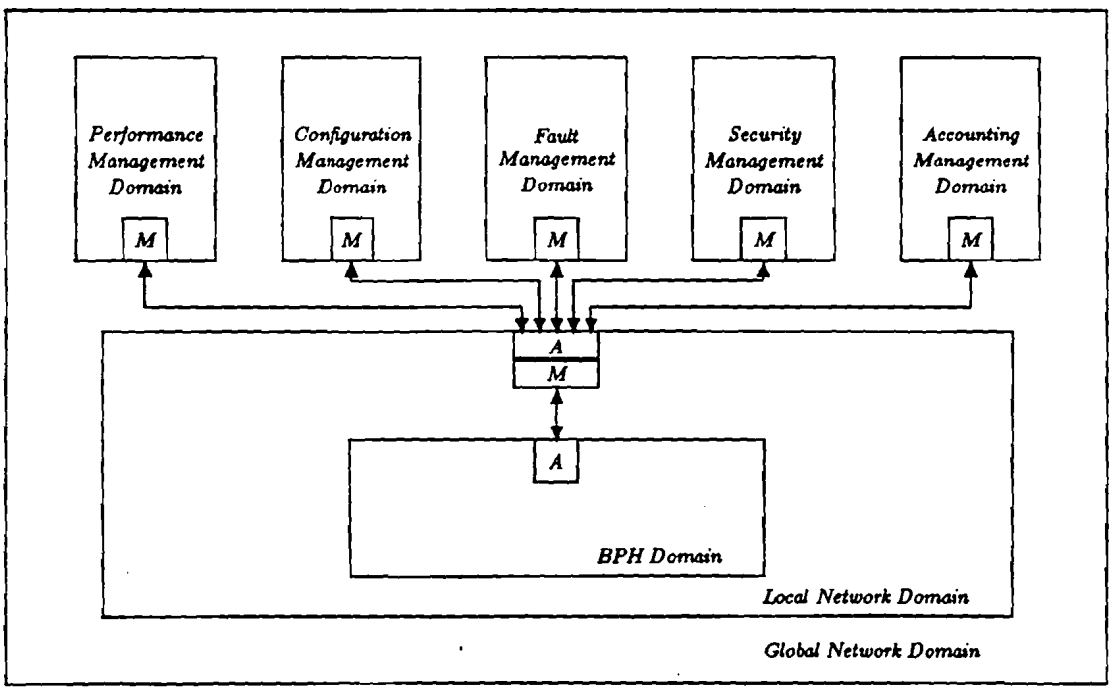



Figure 7.3: *The structure of the network, management view*

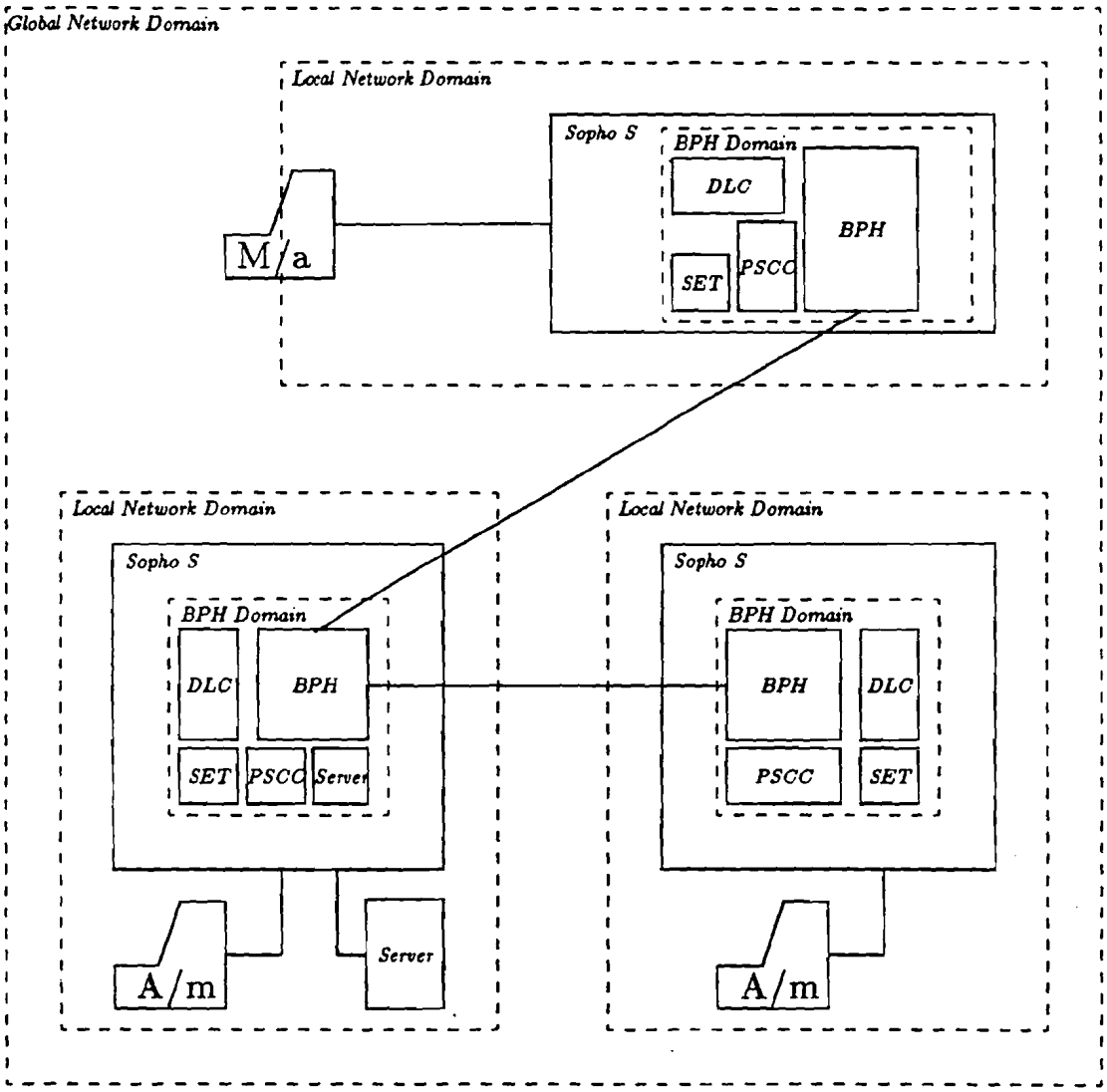
When defining the architecture for management, a hierarchical decomposition of the network can be made. First of all there is the *Global Network Domain (GND).* This is the domain which can contain several Sopho switches. As second domain the *Local Network Domain (LND)* is present. This domain contains the network of BPHs in one Sopho

switch. Note that when there is only one switch, only one LND is present in the global domain, so a manager in this domain does only have one agent. Finally there is the *B-channel Packet Handler Domain (BPHD)*, which consists of a BPH containing the resources, and all the relevant resources needed to provide the packetswitched service with the B-channel Packet Handler (e.g.DLC). This decomposition only takes into account, the physical aspects of the network.

It is also necessary to supply functional domains, in order to be able to perform management functions. These domains should be part of the Global Network Domain, because they will be the actual managers, which have the responsibility for their own domain. They should be possible to access any resource in the network. So the Global Network Domain is functionally decomposited in these five domains (see figure 7.3).
This functional decomposition does not imply however that every action performed on a resource, is done in one functional domain. The correcting of a certain parameter in a protocol entity can be a task of configuration management and performance management for example.

The derived architecture should be realized in the Sopho S environment. A possible way of realizing this architecture is shown in figure 7.4. As is shown in this figure, every switch has a terminal connected to it which is responsible for management of this switch. Every terminal is part of the Local Network Domain, and acts as an agent. One terminal which is part of the Global Network Domain, is the actual management terminal. This terminal acts as manager for the total domain. It is also an agent for its own switch, which is modelled by putting half of the terminal outside this local domain. The architecture depicted is as needed for the management of the packetswitched network service. This implies that the boxes denoted with *set* for example, imply the management of the communication entities from the set as needed for the packetswitched service. So general management functions, not pertaining to packetswitching are not taken into account in this model. The *Packet Switching Call Control* is not needed when the first service (Permanent Virtual Circuits) is implemented. When Switched Virtual Circuits will be available, call control is needed. This call control is modelled by the PSCC block. The definition of this packet switching call control is for further study.

In the domains several resources (BPH,DLC) are present which need to be managed. How these can be defined for management will be treated in the next section.




Note : PSCC is Packet Switching Call Control

Figure 7.4: *The architecture for management in Sopho S environment*

## 7.4 The information model

### 7.4.1 Introduction

A BPH network can be seen as a collection of physical and logical resources. At a given time, the network is in a certain state which is defined by the set of properties of the resources and the relations between these resources. Because the operating of the network is dynamic, the state of the network is constantly changing. What would must be achieved with the management of the BPH network, is that it must be possible to influence the state of the network, or that information about this state can be retrieved. It is therefore important to define what information is relevant for management activities and what resources need to be managed in order to influence the state of the network. Defining the required management information can be separated for each functional domain. The definition of the tasks of these domains will automatically define the information needed from the network in order to be able to do proper management.

OSI uses the *object-oriented* approach to define the information model [89]. Object oriented designing is characterized by the definition of objects. An object is an abstraction of a physical or logical resource. This approach for specifying network management, will also be used for the modelling of the BPH network management activities. The model which is derived in this section must be general enough to model future changes to the BPH network, but it must also be able with the model to give a good definition of the resources present in the network as seen by management.

As stated before, the object oriented approach will be used. In this case the term *managed object* is used.

> A managed object is an abstraction of a physical or logical resource.

Resources which are not defined as managed objects simply don't exist for management, even if these resources are physically present (resources which do not offer an interface to management can be an example of these). Each managed object has a specific managed object class.

> A managed object class is a generic classification shared by a set of similar managed object instances that have similar properties and fulfill the same purposes.

These objectclasses relate to each other in a hierarchical way. This hierarchy is defined by the architecture of the network. When defining

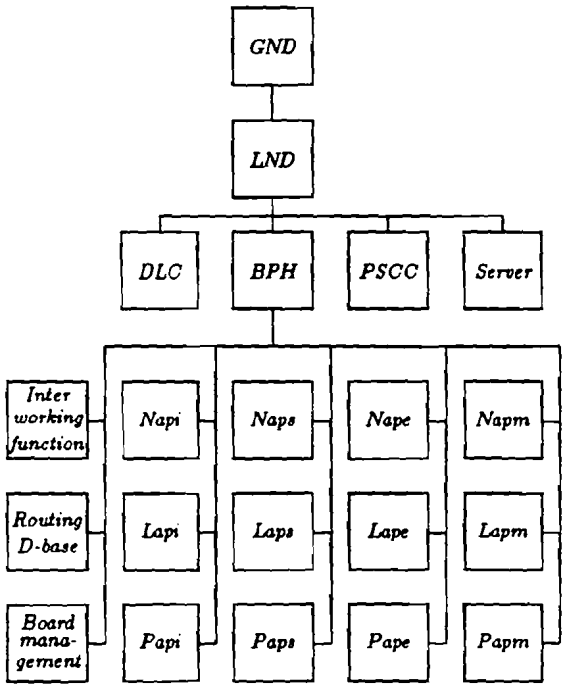



Figure 7.5: *The hierarchical relation tree of managed object classes*

managed object classes the problem is to define several general classifications which can represent the different managed objects. The way to come to this definition of managed object classes is by stepwise refinement.

At the top in the hierarchy there is the managed object class *Global Network.* This object class describes properties, which are valid for every managed object belonging to the packetswitched network. This object class contains a *local network* object class, which describes the properties of the networks in a switch. A local network object class contains several object classes.

**DLC** : This object class describes the relevant items of Digital Line Circuits relating to packet switching. The different SAPIs, needed to route packets coming from sets is an example of such information.

**PSPC** : Object class describing the relation between the Call Control needed for Switched Virtual Circuits, and the BPH. This is for further study.

**Server** : The communicating entities for the packetswitched service in the server, are defined by this object class.

**BPH** : This is the general classification of a BPH board as seen from management. This object class falls apart into several object classes, which describe some specific resource on the BPH-board (see figure 7.5). The naming of these object classes strongly relates to the function performed by the managed objects belonging to the class.

All these object classes contain managed objects, which inherit the properties defined for the object class. In the next section a method to describe these managed object classes will be given.

### 7.4.2 Specification of managed object classes

After having identified which managed object classes are present in the network, a means to specify the managed object classes will now be given. The method which is used to describe a managed object class is currently under study by the ISO/OSI management forum [ref.]. The definitions derived in this study group will also be used for describing the managed object classes in the BPH network. A special notation is provided in order to have a uniform way of writing down the different specifications. Use will be made of a so called *template* [89].

> A template is a high level description, that does not include syntactic detail and can be viewed as a form to be filled in by the designer.

Once the different templates have been specified, it is just a matter of 'filling in' the template specification, for every object to which the template applies. The template for management object classes is as shown in figure 7.6. First a description is given of the different clauses which are mentioned in the template.

*label* : is a name which uniquely identifies the managed object class

DERIVED FROM : *superclass* is a label of the managed object class which is directly above the managed object class in the hierarchy. The DERIVED FROM clause causes the managed object class to inherit all the static and dynamic information from its parent class

| | |
|---|---|
| *label* MANAGED-OBJECT-CLASS | |
| DERIVED FROM | {*superclass*} |
| BEHAVIOUR | |
| DEFINITIONS | {*description of behaviour*} |
| CHARACTERIZED BY | |
| ATTRIBUTES | |
| MUST CONTAIN | {*attributelist*} |
| MAY CONTAIN | {*attributelist*} |
| OPERATIONS | |
| CREATE | |
| DELETE | |
| ACTIONS | {*actionlist* } |
| NOTIFICATIONS | {*notificationlist*} |
| ::= {*object-id*} | |

Figure 7.6: *The managed object class template*

BEHAVIOUR : provides a complete definition of the behaviour of the managed object class. It defines the conditions when a value of an attribute may be changed by management and by the managed object itself. Any notification which results from a particular value of the attribute should also be specified.

CHARACTERIZED BY : contains two subclauses, ATTRIBUTES and OPERATIONS

ATTRIBUTES : attributes consist of an attribute identifier and a value. Attributes are of a certain datatype and are relevant for the managed object in that they contain relevant information for management. The clause contains two lists of attributes. *Mandatory* attributes are attributes which must be present and are summarized within the MUST CONTAIN clause. *Optional* attributes are summarized within the MAY CONTAIN clause.
The access constraint qualifiers, READ-ONLY or WRITE-ONLY may be specified for each attribute

OPERATIONS : contains a list of operations which managed objects may perform or which may be performed on managed objects.

*object-id* : the way in which the class will be referenced by the system

An example of a filled in object class template is shown in figure 7.7. In this example the management object class Lapi is defined. This description defines the properties which all managed objects belonging

| | |
|---|---|
| Lapi MANAGED-OBJECT-CLASS | |
| DERIVED FROM | *{bph}* |
| BEHAVIOUR | |
| DEFINITIONS | *see section about protocol, layer 2* |
| CHARACTERIZED BY | |
| ATTRIBUTES | |
| MUST CONTAIN | *{windowsize,timeoutperiod,n200,fcs}* |
| OPERATIONS | |
| CREATE | |
| DELETE | |
| ACTIONS | *{change-attributes,dl-est-req,dl-rel-req set-own-rec-busy,clear-own-rec-busy}* |
| NOTIFICATIONS | *{dl-error-ind, dl-est-ind,dl-rel-ind}* |
| ::= *{objectclass8}* | |

Figure 7.7: *Example of a filled-in template*

to this class (thus all Lapi entities) will possess. This means that every Lapi entity delivers a minimal interface towards management as defined in the template. The description of the managed objects may add some more specific properties[1]. With a description of the managed object classes in this way, it is easy to see what characteristic items each managed object, belonging to the class, has.

### 7.4.3 The naming of managed objects

As shown in figure 7.5, there is a relation between the managed object classes. This relation can be used for the naming of the managed objects. A definition for this naming is given with the *namebinding template*. The namebinding template allows alternative naming structures to be defined for a managed object class. It is possible for a managed object class to have a relation with more than one other managed object class. This can be modelled by defining more namebinding templates. With this template a definition can be given of the naming, which is used to access a particular managed object. The layout of the namebinding template is shown in figure 7.8.

OBJECT CLASS : defines the object class which is being named

IS NAMED BY : defines the managed object class, instances of which may contain instances of the managed object class being named.

[1] like how they relate to other entities for example.

| | |
|---|---|
| *label-nb* NAME-BINDING | |
| OBJECT CLASS | *managed object class* |
| IS NAMED BY | *managed object class* |
| WITH ATTRIBUTE | *attribute* |
| ::= {*namebinding-id*} | |

Figure 7.8: *The namebinding template*

WITH ATTRIBUTE : contains the attribute which is used for the naming of this managed object class when using this namebinding

An example is given in figure 7.9. This example shows that managed

| | |
|---|---|
| *Lapi-nb* NAME-BINDING | |
| OBJECT CLASS | *Lapi* |
| IS NAMED BY | *BPH* |
| WITH ATTRIBUTE | *Lapi-id* |
| ::= {*namebinding1*} | |

Figure 7.9: *Example of a filled-in namebinding template*

object BPH refers to a Lapi managed object by a unique Lapi-id (e.g. a port#).

With this description of the management for the network, a good overview of which resources need to be managed and how they can be managed, is obtained. For proper management however it must also be realized that managed objects can relate with others. Performing a management action on a managed object, can influence the working of another managed object. These relations will be dealt with in the next section about configuration management.

## 7.5 Configuration management

As was stated before, configuration management deals with the definition, collection, use and management of the configuration data. For management a configuration of a network can be defined by means of the managed objects, and how these objects relate. Thus apart from the definition of the managed objects also the relations between the objects have to be defined. Four different relations can be defined.

1. Relation between BPH boards. This relation defines the routing information needed for the network.

2. Relation between two managed objects of the same type, and which are capable of providing an equivalent service. In case of failure of a managed object, the object denoted by the relation can be used to take over the functions.

3. Relation between service user and service provider managed objects. The relation between a layer 2 and layer 3 entity, is an example of such a relation.

4. Relation between two peer to peer managed objects.

For configuration management these relations are important. Therefore each managed object should have the capability to define the relations it has, with other managed objects.

A managed object has two different states it can be in, as seen from management. These states are *out-service* and *in-service*.

**out-service** : In this state the managed object can not be used. It is currently being utilized by management, for some reason. Two substates can be identified.

1. non-operational, the managed object is not able to operate (e.g. a failure or absent).
2. operational, the managed object is able to operate.

**in-service** : When the managed object is in the in-service state it is available for usage. In this state also two substates are defined.

1. non-operational, the managed object is not able to operate (e.g. no synchronization with peer yet).
2. operational, the managed object is able to operate.

In figure 7.10 these states are depicted. Transitions between the in-service and out-service state can only be performed by management intervention. State transitions from operational to non-operational are caused by operation of the managed object. These transitions should be notified to configuration management, because its information should represent the actual state of the managed object. The different facilities which the configuration management of the BPH network performs, will now be treated [86].

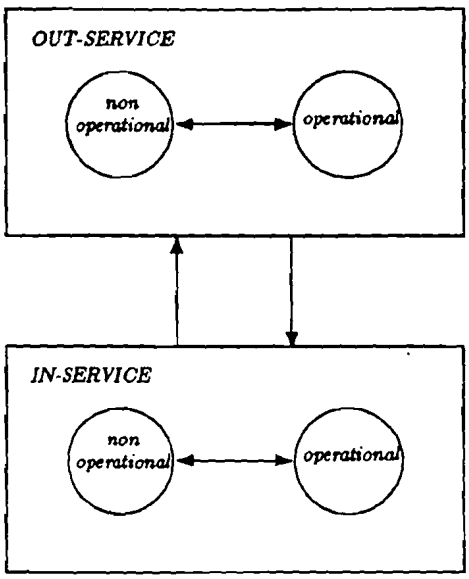



Figure 7.10: *The management states of a managed object*

**Object Configuration Facility :** With this facility it is possible to perform actions on the managed objects, which are valid for that managed object. Creation and deletion of managed objects are the most important actions. Lets consider the adding of a BPH as an example. For configuration management the BPH is a managed object which has to be created. When the object is created, it is in the *out-service* state. The creation of this object also implies that all the relations of this object, the BPH, has to be defined. This means that all the BPHs which interact with the new BPH have to be informed about the appearance of a new BPH. This information has to be stored in the relation field of these managed objects. When adding a new BPH, also a lot of other managed objects (see figure 7.5 for these classes), need to be created before the managed object BPH, can enter the *in-service* state.

When the network of BPHs is being configured, all the managed objects have to created, and the relations established. An object can only enter the in-service state when all the managed objects it relates to, are in-service.

**Attribute Management Facility :** This facility is invoked when actions should be taken on attributes of managed objects. Depending on the attribute, the value may be read or changed. When

a value is changed by normal protocol operation, than it should be notified to configuration management. The changing of the windowsize of a certain Lapi entity, is an example of this facility.

**State Management Facility :** When the management state of an object should change, this facility is invoked. The changing of the states of the managed object when it is operating in a normal way, should not be allowed, because this causes a very cloudy state transition scheme. Only the state transitions between out-service and in-service may be performed by this facility.

**Relationship Management Facility :** The examination, setting and changing of relations between managed objects can be performed with this facility. As noted in the beginning of this section there are four different kind of relations, which all require their specific actions. When a server is added for example, this facility can be used to update the routing information.

As was shown in this section, configuration management is needed to perform a lot of tasks. It provides facilities, which can be used by other functional domains (e.g.Fault Management), to perform their tasks. The formal description of the management of the BPH network allows a structured approach for defining the different functional domains, and thus the management of the network.

# Chapter 8

# Conclusions and Recommendations

## 8.1 About the defined network service

### 8.1.1 The single node BPH service

The B-channel Packet Handler, allows the integration of a packetswitched service into Sopho S. This service extends the functionality of the switch, and can therefore become more attractive to users. The concept which is based on Permanent Virtual Circuit access, is intended for primaly use of the BPH service. Because only a single node solution is offered there are some limitations to this service.

1. No more than 240 sets can access the BPH service.
2. Users of the BPH service need to be in one Sopho switch.
3. Up to four servers can be connected to the BPH.

To overcome these limitations, and to be able to offer a more flexible service, a networking service with the BPH concept is defined. The defined packetswitched network service in this report allows a flexible approach for networking with the BPH concept as it was defined.

### 8.1.2 About the network service characteristics

The following can be concluded from the definition of this network service.

- The network service puts no restrictions on the network access protocols. It allows a migration towards the provision of standard services like X.31 or I.122 services. These services need the introduction of new access protocols, and have no impact on the network service as defined.

- The service is independent of the application offered by a possible server. This also means that it is not necessary that a server is present. An application may also be running on a set, without any server interaction. This could be used for example to transfer messages from set to set.

- The network service allows network wide D-channel access of a set. No restrictions are put on the location of the sets, meaning that the server does not have to be part of the switch to which the set connects.

- A flexible approach is offered to connect servers to the BPH network. The architecture derived allows servers to attach to the network, at any location. It is advised to offer several application within one server, in order to minimize the number of servers. This greatly reduces the complexity of the topology design.

### 8.1.3 About the architecture

In order to meet all the requirements demanded from the network a reliable and flexible architecture is needed. Two possible architectures are derived. Depending on the situation one of them suits best.

1. Topology 1 is a centralized star topology. This topology serves a maximum number of sets, while keeping into account that no more than three BPHs are in the path from set to server. It is not robust, in case of a BPH or link failure no alternative routing can be applied. A maximum of four servers can be served with this topology.

2. Topology 2 is also a star topology, but now a fully interconnected network of BPHs exist. By offering more links to interconnect, a more robust topology is derrived that allows alternative routing. This topology also allows a maximum number of servers to access the network, to any BPH present in the fully interconnected network.

Both topologies deliver a flexible architecture which allows users to access the network service without any real physical limitations. It is advized to use a topology which uses the principles derived in this report, but which is maximized for the situation present. A trade off between link costs and needed reliability should be made, when defining the architecture for the network service.

### 8.1.4 About the protocol for interconnection

To be able to transfer information across the network, a protocol is defined for BPH peer-to-peer communication. The protocol offers a proprietary frame switching service. Full layer 2 termination is provided at each BPH and minimal layer 3 intervention. Because no association exists between two communicating BPHs, only routing is performed in layer 3. Layer 2 takes care of a more efficient use of the physical link by means of layer 2 multiplexing (different Service Access Points), and using a sliding window mechanism for information transfer. It delivers two reliable, sequence controlled data links[1] to layer 3. Error correction is obtained by means of the Go Back N algorithm.

The protocol of layer 2 is formally specified in CCITTs Specification and Description Language. SDL is a very powerful specification language for communication protocols.

1. The graphical representation is very well suited for documentation. It can be used as a readable, unambigious specification guide, to be used for implementation.

2. It is easy to model a protocol stack in SDL. With SDL a very structured model can be defined.

3. It is possible to do simulations with the specification. When making a formal specification, a simulator can be generated. This simulator is also generated for layer 2 of the protocol for interconnection. With this simulator the specification has been thoroughly tested. A drawback is that no real time simulations can be performed.

Because of the many protocol stacks already present in the BPH, it is advized to uses the layer 2 protocol for interconnection also for BPH user communication. This reduces the number of different entities present

[1] one for user data and one for management data.

in the BPH. Because this protocol is formally specified and tested, a specification is present which allows easy implementation.

### 8.1.5 About the management of the network service

To be able to offer a reliable network service, proper management facilities are needed. The definition given of the network management, is a good guideline on how to deal with the management problem. By defining the resources as managed objects, a specification is obtained that puts no constraints on the implementation. This specification is nevertheless very usable for implementation of the different resources, with an eye towards management.

A functional and architectural decomposition is made of the network management problem. Five different functional domains are defined : Configuration, Fault, Performance, Accounting and Security management. Configuration management will need most attention for a first implementation. This management domain delivers the facilities to initialize the network and to access the different resources present in the network.

The architecture defined in this report provides a flexible, distributed infrastructure, for management activities. The manager can perform actions on every resource present in the network which delivers a management interface. The interaction takes place via a workstation connected anywhere in the network. This architecture puts the intelligence outside the Sopho S, in a special terminal. This allows a migration towards standard management services, which will become available in the near future.

It is advized to use the concepts derived in this report for the management of the network also for other Sopho S managament domains.

## 8.2 Evolution towards the fully integrated, distributed packetswitched network service

As the packetswitched network service is deployed, new applications will be developed. This means that the services offered by the network will evolve as well. To be able to offer services from the beginning an

evolution is proposed, which eventually allows the migration towards standardized packetswitched services.

**Step 1:** Single node BPH service based on Permanent Virtual Circuits between users and BPH. No more than four servers are attached to this BPH.

**Step 2:** BPH network service, based on Permanent Virtual Circuit between users and BPH. The network may be distributed over several switches. The applications which can be accessed with this network service are the same as with the first service. This steps enlarges the number of users which can access the BPH service.

**Step 3:** Enlarging the range of services offered by the packetswitched network.

1. Access of users of the network service by means of Switched Virtual Circuits. This provides users with a more flexible service.
2. Offering X.31 services based on D-channel access (ISDN virtual circuit bearer services). This means adaption of the network access protocols.
3. Offering I.122 services for D-channel access. It does not seem logical to offer an unreliable frame relaying service for D-channel access. Due to the limited throughput of this channel (16 kbit/s), services which do not need a high throughput are more useful for this channel. The B-channel is a better solution for offering this service. If however a frame relaying service should be offered for D-channel access, the concepts derived in this report are open-ended enough to implement this service with minor changes to the protocol used for interconnection.

## 8.3 For further study

The following items are not defined yet and are for further study.

1. Estimating the performance of the network service. The network service as it is defined is only valuable if the delays are acceptable. Because a model of the BPH is derived, and the architecture of the network is defined, a total model of the network can be made.

For correct calculations information about the performance of a single BPH, and the expected traffic should be obtained.

2. More specific definition of the management functions. In this report only an abstract definition is given. When implementing the network care should be taken of a proper interface to management. The definition of the management functions and the protocol to be used for the transport of management information are for further study.

# List of Abbreviations

| | |
|---|---|
| **SDL:** | Specification and Description Language. |
| **BCS :** | Business Communication Systems |
| **BPH :** | B-channel Packet Handler |
| **CCITT :** | Commitee Consultatif International de Telephonie et de Telegraphie |
| **CCS :** | Calculus of Communicating Systems (Milner, Hoare) |
| **CEPT :** | Conference Europeenne des administrations des Postes et des Telecommunications |
| **CI :** | Communication Interface, chip name |
| **CPU :** | Central Processor Unit |
| **CRC :** | Cyclic Redundancy Check |
| **CSDN :** | Circuit Switched Data Network |
| **DLC :** | Digital Line Circuit |
| **EC :** | Earth Calling signalling, ATU |
| **ECMA :** | European Computer Manufacturers Association |
| **ET :** | Exchange Termination (Q.921) |
| **FCS :** | Frame Check Sequence (Q.921) |
| **HDLC :** | High level Data Link Control, protocol |
| **IEEE :** | Institute of Electrical and Electronic Engineers |
| **IMP :** | Internal Message Protocol |
| **IP :** | Interworking Protocol |
| **ISDN :** | Integrated Services Digital Network |
| **ISO :** | International Standards Organisation |
| **ISPBX :** | Integrated Services PABX |
| **Kbit :** | Kilo bit |
| **LAM :** | Line Adaptor Module |
| **LAN :** | Local Area Network |
| **LAPB :** | Link Access Protocol Balanced |
| **LAPD :** | Link Access Protocol for the D-channel |
| **LR :** | Low Range |
| **Mbit :** | Mega bit |

| | |
|---|---|
| **NAP :** | Network Access Protocol |
| **OSI :** | Open Systems Interconnection |
| **PABX :** | Private Automatic Branche eXchange |
| **PCC :** | Peripheral Circuit Controller, chip name |
| **PCI :** | Protocol Control Information |
| **PCT :** | Peripheral CircuiT |
| **PDU :** | Protocol Data Unit |
| **PPU :** | Peripheral Processor Unit |
| **PSDN :** | Packet Switched Data Network |
| **PSPDN :** | Public Switched Packet Data Network |
| **PSTN :** | Public Switched Telephone Network |
| **PTDSN :** | Philips Telecommunicatie en DataSystemen Nederland |
| **Q.921 :** | I.441, ISDN Layer 2 |
| **RNR :** | Receiver Not Ready (Q.921) |
| **RR :** | Receiver Ready (Q.921) |
| **SABM :** | Set Asynchronous Balanced Mode (Q.921) |
| **SAP :** | Service Access Point (Q.921) |
| **SAPI :** | Service Access Point Identifier (Q.921) |
| **SDL :** | Specification and Description/Design Language from CCITT |
| **SDU :** | Service Data Unit |
| **SN :** | Switching Network in SM |
| **SOPHO :** | Synergetic Open PHilips Office automation |
| **TA :** | Terminal Adaptor (ISDN, DPNSS) |
| **TDS :** | Telecommunicatie en DataSystemen |
| **TE :** | Terminal Equipment (Q.921) |
| **TEI :** | Terminal Endpoint Identifier (Q.921) |
| **TMS :** | Telephony Management System |
| **V24 :** | V24 standard interface defined by CCITT |

# Bibliography

## References Chapter 2

[1] *The OSI ISO reference model.* CCITT recommendations X.200.

[2] Permouth Y. *An introduction to the basic reference model.*

[3] Tanenbaum A.S. *Computer networks.* Prentice Hall, 1981.

## References Chapter 3

[4] CCITT recommendation X.31. Blue book, 1989.

[5] Noppen van P. *Internal communication model.* BCS project documentation, Vol.SR2271-TL-6419, 1989.

[6] Schuring T. *Packet server voor Sopho S.* BCS project documentation, Vol.SB2280-89.175, 1989.

[7] Bocker P.ISDN, das diensteintegrierende digitale nachtrichtennetz., 1988.

[8] Barberis G, Guarneri M, Macrina P. *Handling packet services within ISDN.* Computer communications 1987

[9] Cooper N. *Packet mode services for ISDN.* Br Telecom Technol J Vol 6, no.1 1988.

[10] Frantzen V. *Packet-switched data communication services in ISDN.* Siemens publication, 1989.

[11] Havermans G, Mulder R. *Terminal interfaces in ISDN.* Telecommunications, 1987.

[12] Hiramatsu Y. *ISDN Packet services, present and future.* Tencon 87 (IEEE), 1987.

[13] Lai W. *Packet mode services: from X.25 to frame relaying.* Computer communications, 1989.

[14] Magnolfi G, Pezzotta A, Valbonesi G. *Integrated system evolution towards ISDN for value added services.* Innovative services or innovative technology, 1989.

[15] Purser M. *X.25-the fulcrum for network standardization.* Computer communications, 1988.

[16] Staudinger W. *The evolution of a packet switching concept for the ISDN from the view of CCITT.* Innovative service or innovative technology, 1989.

[17] Turman B. *Mixing packets and ISDN: part II.* Data Communications, July 1988.

[18] Unsoy M. *How packet-mode transmission services will evolve in ISDN.* Data communications, April 1988.

[19] Unsoy M. *ISDN packet services evolution.* IEEE saic, 1987.

[20] Unsoy M, Juskevicius E, Baker D. *Packet switching architecture for ISDN.* ISDN87, 1987.

[21] Fogarty K. *Introduction to the CCITT I series recomendations.* Br Telecom tecnol J Vol.6, 1988.

[22] *Framework for providing additional packet mode bearer services.* CCITT recommendations I.122, 1989.

## References Chapter 4

[23] Havermans G. *Description of Laps.* BCS project documentation, Vol.03:62:02, 1988.

[24] Hoff van 't E.L. *Sopho S BPH-TMS network access protocol.* BCS project documentation, Vol.03:63:xx, 1988.

[25] Hoff van 't E.L. *Packet switching in Sopho S.* BCS project documentation, Vol.SR2271-TL-6435, 1988.

[26] Hoff van 't E.L. *BPH external requirements.* Sopho SLR project documentation, Vol.0400:11:01, 1988.

[27] Klik C.M. *Packet mode access via Sopho S.* BCS project documentation, Vol.0000:58:04, 1988.

[28] Ottink R. *BPH firmware design specification.* BCS project documentation, Vol.SR2271-TL-6067, 1988.

## References Chapter 5

[29] Anderson G.A. *Computer interconnection structures: taxonomy, characteristics, and examples.*

[30] Feng T.Y. *A survey of interconnection networks.* IEEE, 1981.

[31] Green P. *An introduction to network architectures and protocols.* IEEE transactions on communications, Vol.com-28, 1980.

[32] Shoch J.F. *Inter-network naming, addressing and routing.* IEEE, 1978.

[33] Vikram R.S. *Topological analysis of packet networks.* IEEE journal on selected areas in communications, 1989.

[34] Heiber A. *An overview of universal information services:concepts and technologies of future networks.* AT&T technical journal, 1989.

[35] Lynch D.C. *TCP/IP and OSI.* Summer, 1989.

[36] Luetchford J. *CCITT Recommendations-network aspects of the ISDN.* IEEE journal on selected areas in communications vol.sac4, 1986.

[37] Frantzen V. *Trends in the development of public telecommunications networks.* Computer networks and ISDN systems, 1987.

[38] Backes F. *Spanning tree bridges, transparant bridges for the interconnection of LANs.* IEEE Network, Vol.2, 1988.

[39] Dixon R, Pitt D. *Source routing bridges, addressing, bridging and source routing.* IEEE Network Vol.2, 1988.

[40] Benhamou E. *Integrating Bridges and routers in a large internetwork.* IEEE Network Vol.2, 1988.

[41] Salwen H, Boule R, CHiappa J. *Examination of the applicability of router and bridging techniques.* IEEE Networks Vol.2, 1988.

[42] Zhang L. *Comparison of two bridge routing approaches.* IEEE Networks Vol.2, 1988.

[43] Seifert W. *Bridges and routers.* IEEE Network Vol.2, 1988.

[44] *Packetized data transfer in private switching networks.* ECMA TC32-TG6, 1987.

[45] Brown L.M. *Unix system V data networking Architecture.* AT&T, Vol.3, No.3.

[46] Dunogue J. *the building of intelligent networks.* Communication & Transmission no.2, 1989.

[47] Karol M.J. *Using a packet switch for circuit-switched traffic: A Queueing system with periodic input traffic.* IEEE transactions on communications vol.37, 1989.

[48] Shalmon M.S. *Exact delay analysis of packet-switching concentrating networks.* IEEE transactions on communications,vol.com.35, 1987.

[49] Dutta A. *Backbone networks design: knowledge based perturbation methods.* IEEE Vol.CH2503, 1987.

[50] Sahin V. *Ne functions and OSI protocol architectures for operations data networking* IEEE Vol.CH2424, 1987.

[51] Pitt D.A. *Address-based and non-address-based routing schemes for interconnected local area networks* Local area and multiple access networks pp.155-166, 1986.

[52] Liebl F. *Routing-algorithmen fur packetvermittelende datennetze-ein beitrag zur systematik.* NtzArchiv Bd 9., 1987.

[53] Saksema V.R. *Analysis of routing issues in the design of packet networks.* IFAC large scale systems, 1986.

[54] Smith P. *Adressing a problem with packet switching.* Communications International, 1989.

[55] Tucker J.B. *Naming & addressing for interworking.* Networks, 1985.

[56] Carparelli O. *Internetworking between a private packet mode ISDN and a PSDN.* ISDN, 1987.

[57] Turman B. *Bell operating company packet interfaces between networks and subnets.* Computer networks and ISDN systems, 1988/89.

[58] Lai W.S. *Network and nodal architectures for the interworking between frame relaying services.* Computer Communications, 1989.

## References Chapter 6

[59] C.J. Koomen. *System Technology.* Study notes for college, 1988.

[60] Course material on High-level Data Link Control protocol. Philips internal course.

[61] *SGE/PC user manual.* Telelogic AB, 1989.

[62] Belina F. *The CCITT-specification and description language SDL.* Computer networks and ISDN systems, 1989.

[63] Verhoosel J. *A summary of the specification and description language.* No reference, 1989.

[64] Hogrefe D. *OSI service specification with CCITT SDL.*

[65] Chen P. *How to make the most of ISDN's new LAPD protocol.* Data communications, 1987.

[66] Moughton J, Evans B, White D. *An experimental ISDN X.25 adaptor.* ISDN 87, 1987.

[67] Burg F, Puges P. *X.25: It's come a long way.* Computer networks and ISDN 1989.

[68] Folts H. *X.25 Transaction-oriented features-Datagram and Fast select.* IEEE transactions on communications, Vol.com-2, 1980.

[69] ECMA Standard-105. *Datalink layer protocol for the D-channel of the S-interfaces.* 2nd Edition 1987.

[70] Rybczynski A. *X.25 interface and end-to-end virtual circuit service characteristrics.* IEEE transactions on communications vol.com-28, 1980.

[71] Rybczynski M, Palframan J. *Common X.25 Interface to public data networks.* Computer networks, 1987.

[72] Sloman M. *X.25 explained.* Computer communications, 1978.

[73] *Functional specification of the protocol coprocessor.* Philips internal document, Vol.20.11.3.0, 1988.

[74] Gunningberg P. *Application Protocols and performance benchmarks.* IEEE communications magazine, 1989.

[75] Brady P.T. *Performance of an edge-to-edge protocol in an simulated x.25/x.75 packet network.* IEEE journal oon selected areas in communications vol.6, 1988.

[76] Coackley R. *The ubiquitous protocol analyzer* Telephony, 1989.

[77] Lombardo A. *An Extended algebra for the validation of communication protocols.* Software engineering journal, 1989.

## References Chapter 7

[78] Morell B.E. *Building a management architecture for network control.* Businees communications rewiew, 1987.

[79] Flanagan W.A. *A case against distributed architecture.* TE&M, 1988.

[80] Kiesel W.M. *Netwerkmanagement van industriele lokale netwerken.* PT Electonica-electrotechniek, 1989.

[81] Burvill M. *Improving management with packet switches.* Communications International, 1989.

[82] Howard F. *The real network management problem.* Business communication review, 1988.

[83] Terplan K. *Network management evolution.* Computer communications, 1988.

[84] Hannon D.C. *Network management for the BCPN.* RACE 1011 (BCPN), 1990.

[85] *Information retrieval, transfer and management for OSI.* ISO/IEC DP 10165-1, 1989.

[86] *Configuration Management Service Definition.* ECMA, TC32/87/285, 1987.

[87] *User-network management.* CCITT recommendation Q.940, 1990.

[88] *Framework for OSI management.* ECMA TR/37, 1987.

[89] *Forum Object Specification Framework.* OSI network management forum, 1989.

[90] *Generic management model for private telecommunication networks.* ECMA technical report, 1990.

[91] *Information retrieval, transfer and mnanagement for OSI.* ISO/IEC DP 10164-1, 1989.

[92] *Systems management overview.* ISO/IEC DP 10040, 1990.

[93] Klerer S.M. *The OSI management architecture : an overview.* IEEE network, 1988.

[94] Widl W. *Standardization of telecommunication management networks.* ERICSSON review no.1, 1988.

[95] Sluman Ch. *A tutorial on OSI management.* Computer networks and ISDN systems 17, 1989.

[96] Wright R. *network management-The open systems future.* Telecommunications, 1988.

[97] *Administration in an OSI framework.* SPAG, 1989.

[98] Hird E.V. *network management with supermasters.* TE&M, 1988.

[99] Terplan K. *Network management evolution.* Computer communications, vol.11 no5, 1988.

[100] Bennet G. *Simple solutions to managing networks.* Communications, 1989.

[101] Frank H. *The real network management problem.* Business communications review, 1988.

[102] Anderson H. *Network management: the future is now.* The yankee group Europe, 1988.

[103] Cheshire P. *Map&Top in perspective: the role of communications in integration.* networks, 1987.

[104] *Netwerkbeheer nog in de kinderschoenen.* Pcnetwerk, 1989.

[105] Gale T. *Community management: an approach to managing distributed systems.* networks, 1987.

[106] Moretti D. *An architecture for LAN management.* Networks, 1987.

[107] Fletcher J.H. *Human factors engineering in network management.* Telecom technol J Vol.6, 1988.

[108] *Network management systems.* Networks, 1988.

[109] Kiel F. *Networks management systems.* Electrial communication vol.62, 1988.

[110] *DEC steers new course toward OSI net management.* Data Communications, 1988.

[111] Sluman Ch. *Network and systems management in OSI.* Telecommunications, 1988.

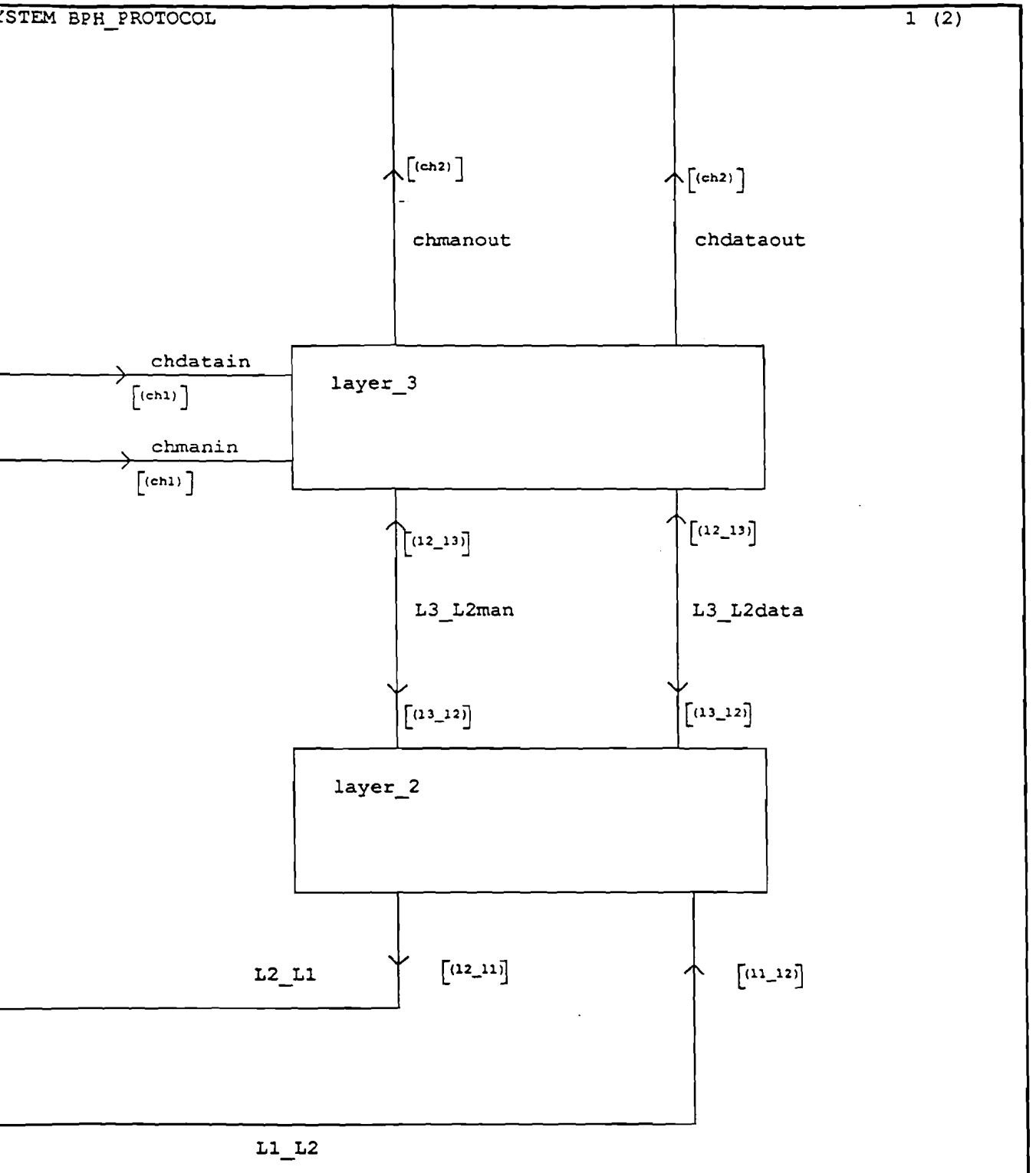
[112] Cowin S. *Network management: yesterday, today and tomorrow.* data communications, 1987.

[113] Huntington J. *OSI-based net management: is it too early, or too late?.* Data Communications International, 1989.

[114] MacInnes A. *Software products framework for diagnosing network problems.*

# Appendix A

# SDL description of the protocol for interconnection


YSTEM BPH_PROTOCOL
1 (2)
[ch2]
[ch2]
chmanout
chdataout
chdatain
[ch1]
layer_3
chmanin
[ch1]
[12_13]
[12_13]
L3_L2man
L3_L2data
[13_12]
[13_12]
layer_2
L2_L1
[12_11]
[11_12]
L1_L2

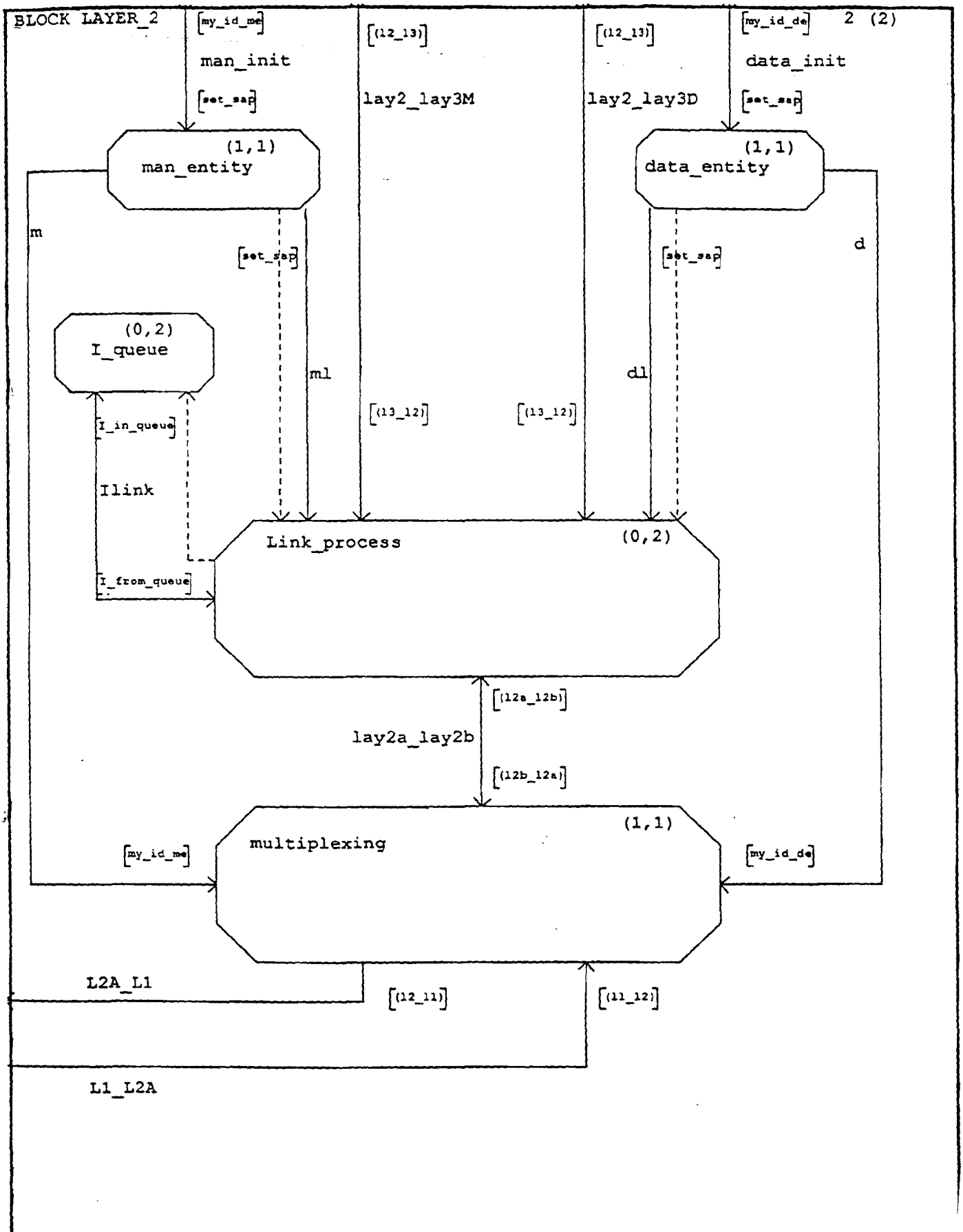
SYSTEM BPH_PROTOCOL 2 (2)

```
newtype datagram_type
struct
    dest_adr  integer;
    sour_adr  integer;
    data      charstring;
endnewtype datagram_type;

newtype control_type
literals
    I, RR, RNR, SABM, UA
operators
    ordering
endnewtype control_type;

newtype flag_type
literals
    01111110;
operators
    ordering
endnewtype flag_type;

newtype sapi_type
literals
    15, 63; /* 15=data, 63=management */
operators
    ordering
endnewtype sapi_type;

newtype control_record_type
struct
    kind      control_type;
    seq#      integer;
    data      datagram_type;
endnewtype control_record_type;

newtype frame_type
struct
    flag      flag_type;
    sapi      sapi_type;
    control   control_record_type;
    fcs       integer;
endnewtype frame_type;

newtype man_contr_type
literals
    s_o_r_b,c_o_r_b;
operators
    ordering
endnewtype man_contr_type;

newtype man_err_type
literals
dl_err;
operators
    ordering
endnewtype man_err_type;

SIGNAL
set_sap(PId),my_id_me(PId),my_id_de(PId),
transf_data(datagram_type), data_arrived(datagram_type),
dl_est_req, dl_est_ind, dl_rel_ind, .
dl_data_req(datagram_type),dl_data_ind(datagram_type),
ph_data_req(frame_type), ph_data_ind(frame_type),
mdl_err_ind(man_err_type),mdl_contr_req(man_contr_type);
signallist ch1=transf_data,mdl_contr_req;
signallist ch2=data_arrived, mdl_err_ind;
signallist l3_l2=dl_est_req,dl_data_req,mdl_contr_req,set_sap;
signallist l2_l3=dl_est_ind, dl_rel_ind,dl_data_ind, mdl_err_ind,my_id_me,
          my_id_de;
signallist l2_l1=ph_data_req;
signallist l1_l2=ph_data_ind;
```

OCK LAYER_2 1 (2)

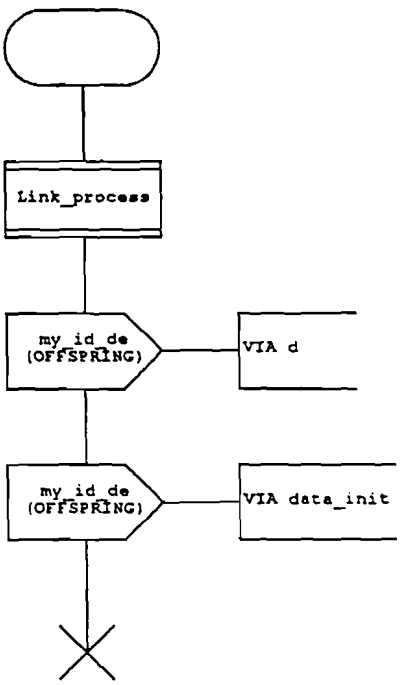
```
connect L2_L1 and L2A_L1;
connect L1_L2 and L1_L2A;
connect L3_L2man and lay2_lay3M, man_init;
connect L3_L2data and lay2_lay3D, data_init;
```

```
SIGNAL
I_in_queue,I_from_queue,
frame_received(control_record_type), transmit_frame(control_record_type);
signallist l2b_l2a=transmit_frame;
signallist l2a_l2b=frame_received;
```


BLOCK LAYER_2
2 (2)
[my_id_me]
man_init
[set_sap]
[12_13]
lay2_lay3M
[12_13]
lay2_lay3D
[my_id_de]
data_init
[set_sap]
(1,1)
man_entity
(1,1)
data_entity
m
d
[set_sap]
[set_sap]
(0,2)
I_queue
ml
dl
[I_in_queue]
Ilink
[I_from_queue]
[13_12]
[13_12]
Link_process
(0,2)
[12a_12b]
lay2a_lay2b
[12b_12a]
(1,1)
multiplexing
[my_id_me]
[my_id_de]
L2A_L1
[12_11]
[11_12]
L1_L2A

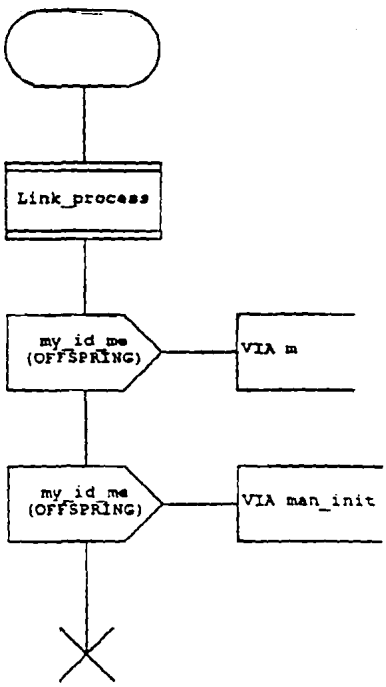
ROCESS DATA_ENTITY 1 (1)

This process creates a
nk process which is to
used for the transfer of
ta. It distributes the
ocess-id to
Process Napidata (layer 3)
Multiplex process (layer 2A)



PROCESS MAN_ENTITY 1 (1)

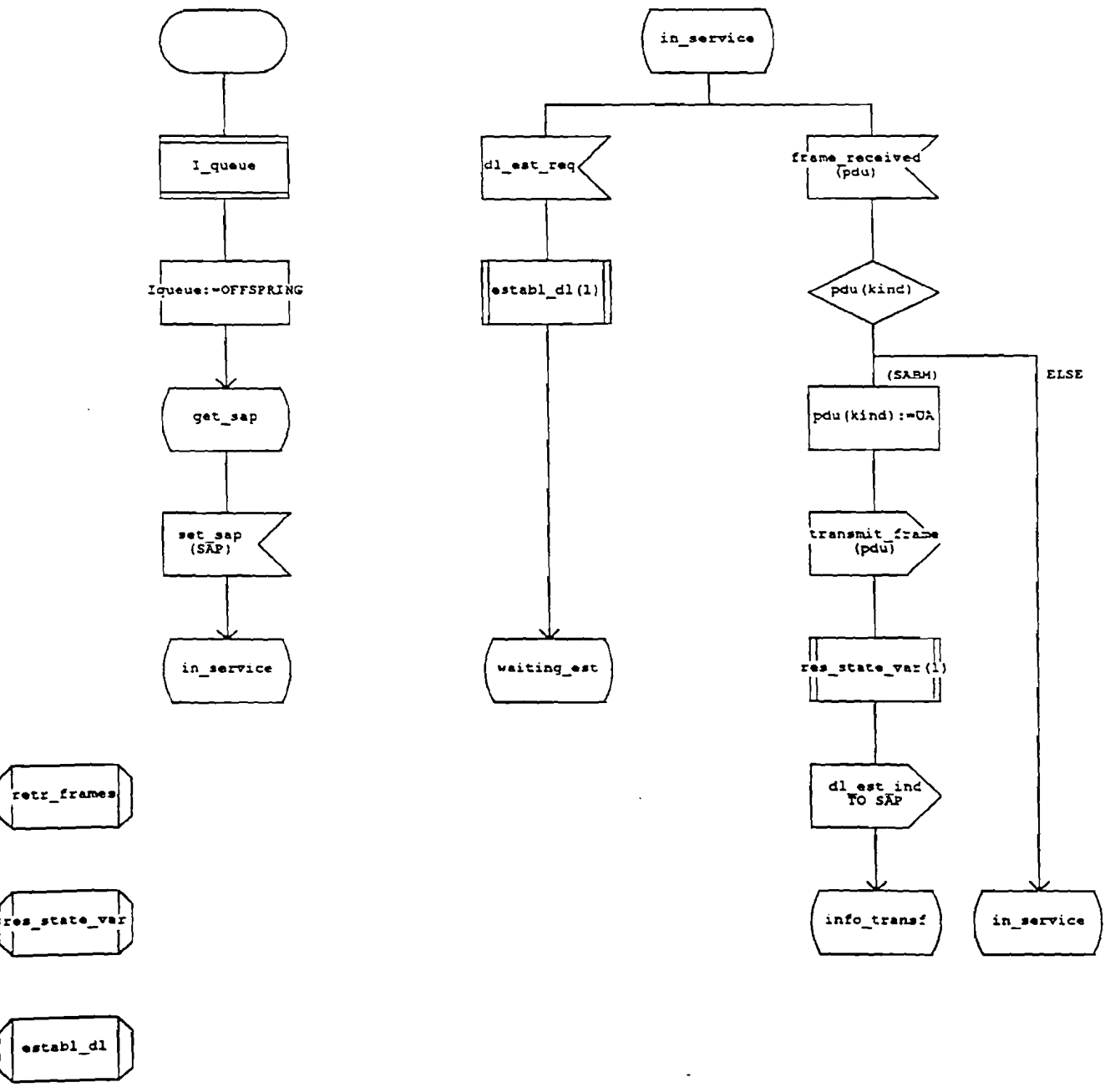
```
/* This process creates a linkprocess
which is used for the transfer of
management information. It sents the
processid to
1.Napimanagement (layer 3)
2.Multiplexing process (layer 2A)
```



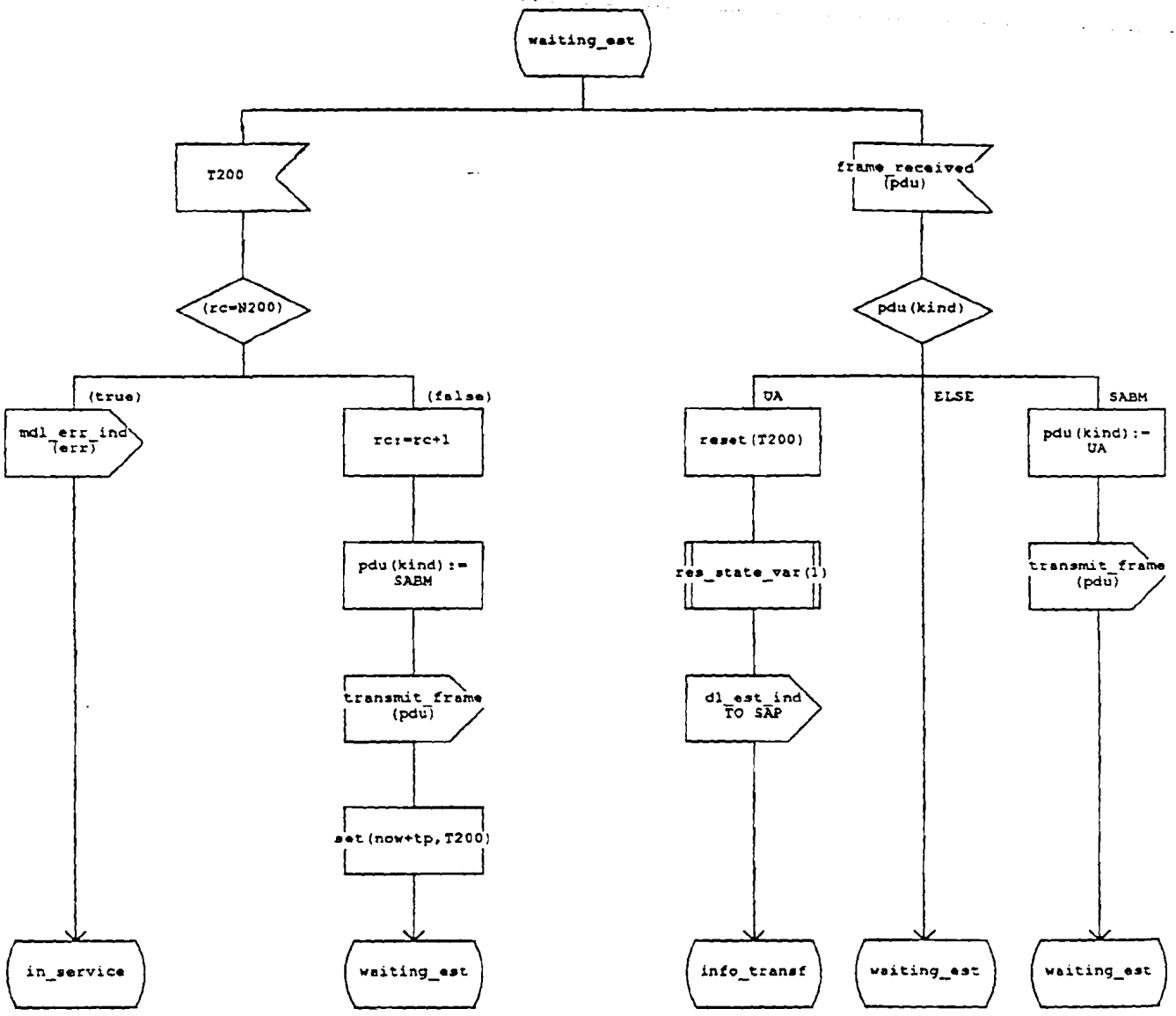
ROCESS LINK_PROCESS 1 (10)

This process takes care of
e peer to peer protocol for
o layer 2 entities. It uses
ocess I-queue for the storing
I-frames. It receives the SAP
lue to which it should sent the
ceived layer 3 data */

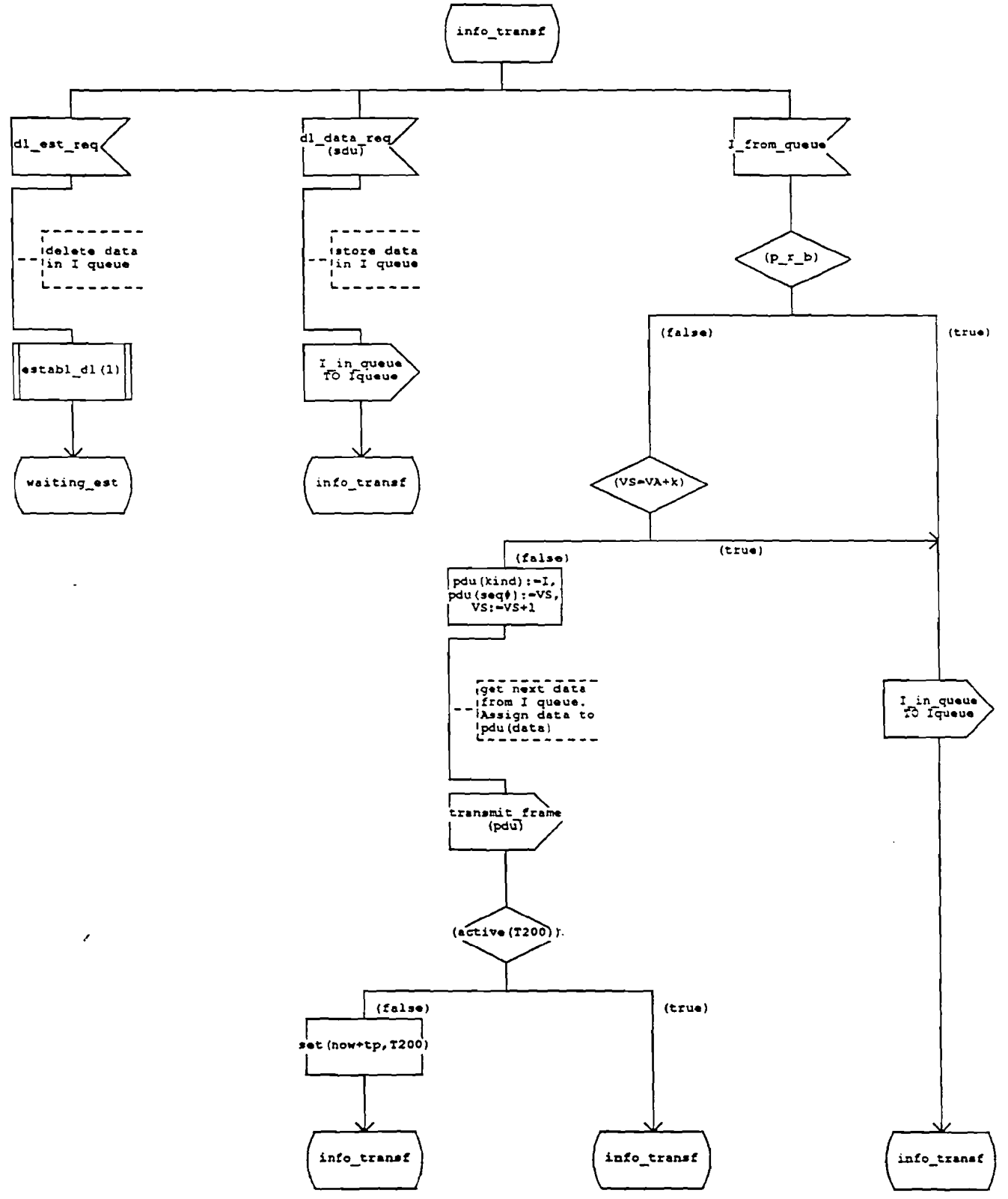
```
timer T200;
dcl Iqueue,SAP PId,
VS,VA,VR,NR,NS,rc integer,
o_r_b, p_r_b boolean,
pdu control_record_type,
sdu datagram_type,
err man_err_type,
contr man_contr_type;
synonym N200 integer = 3;
synonym k integer = 8;
synonym tp duration = 2;
```



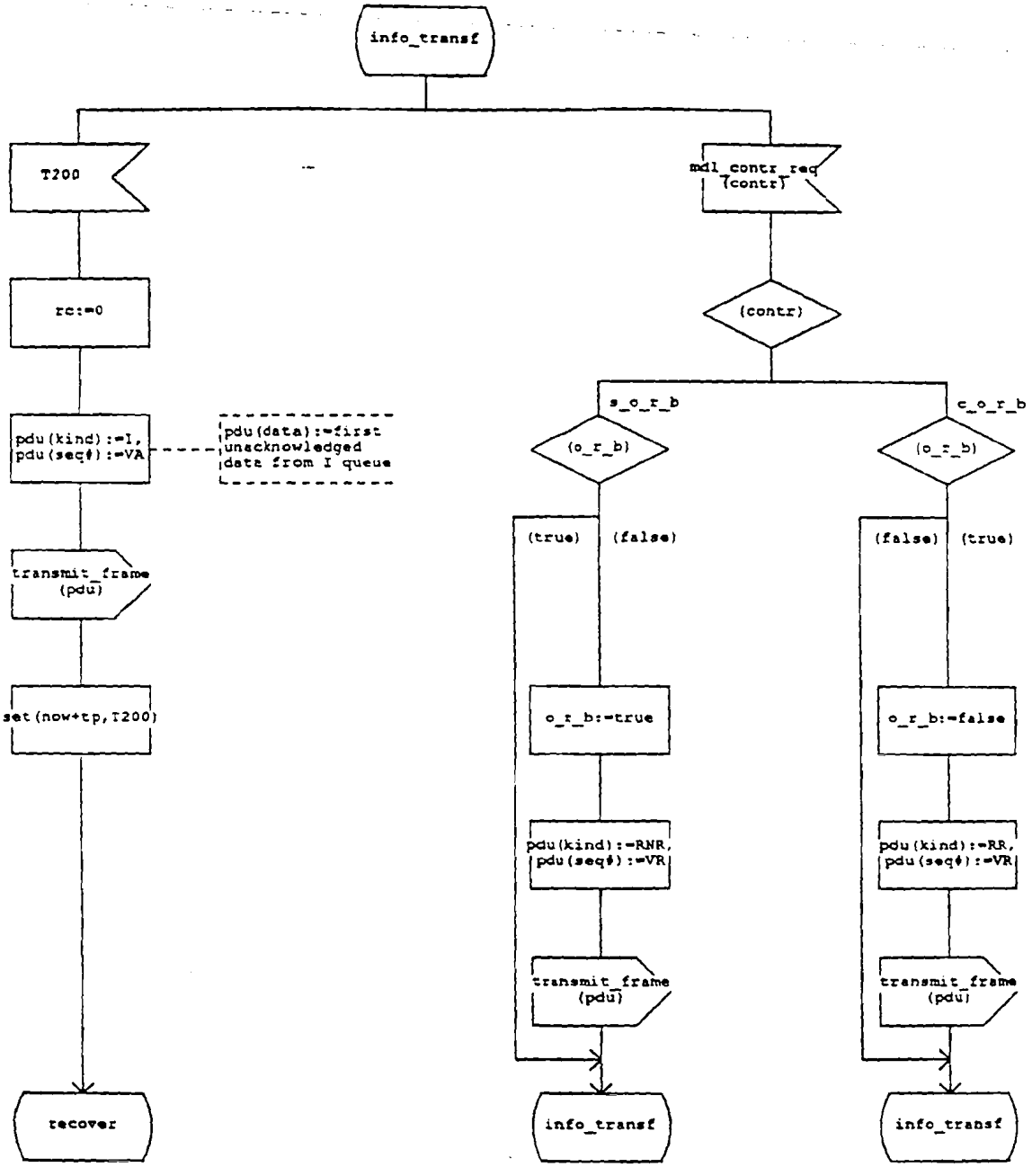
PROCESS LINK_PROCESS 2 (10)



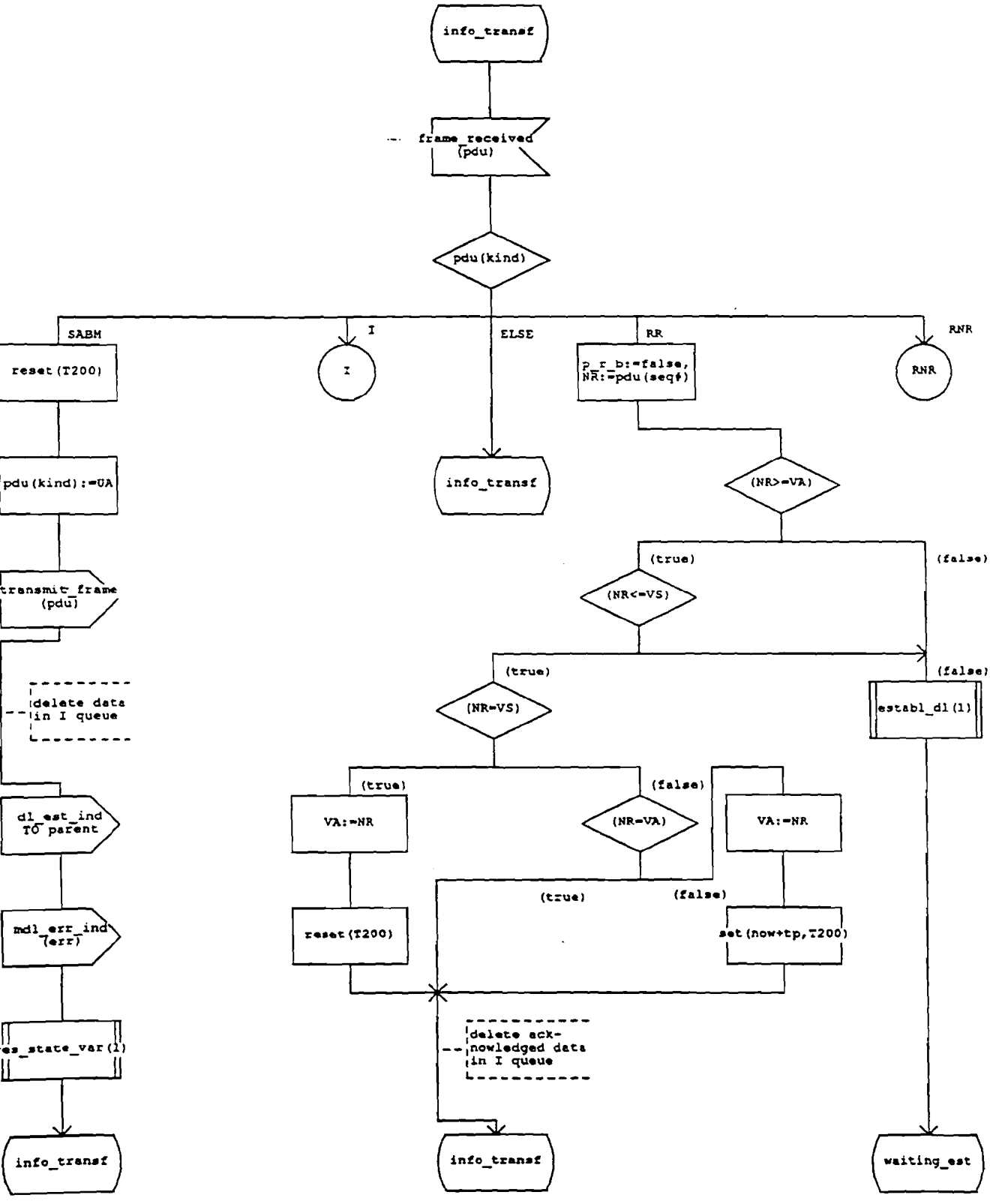
PROCESS LINK_PROCESS 3 (10)



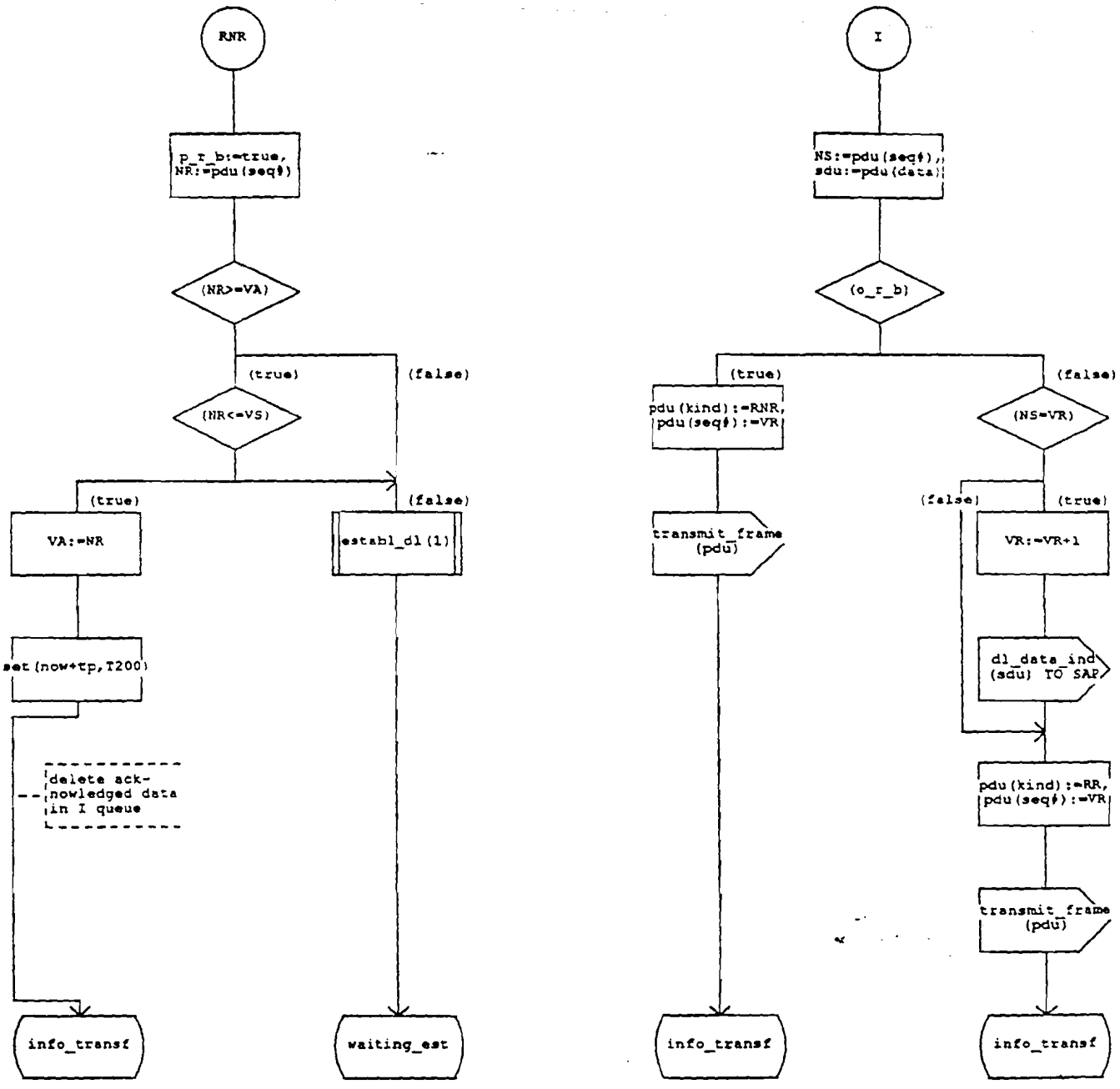
PROCESS LINK_PROCESS 4 (10)



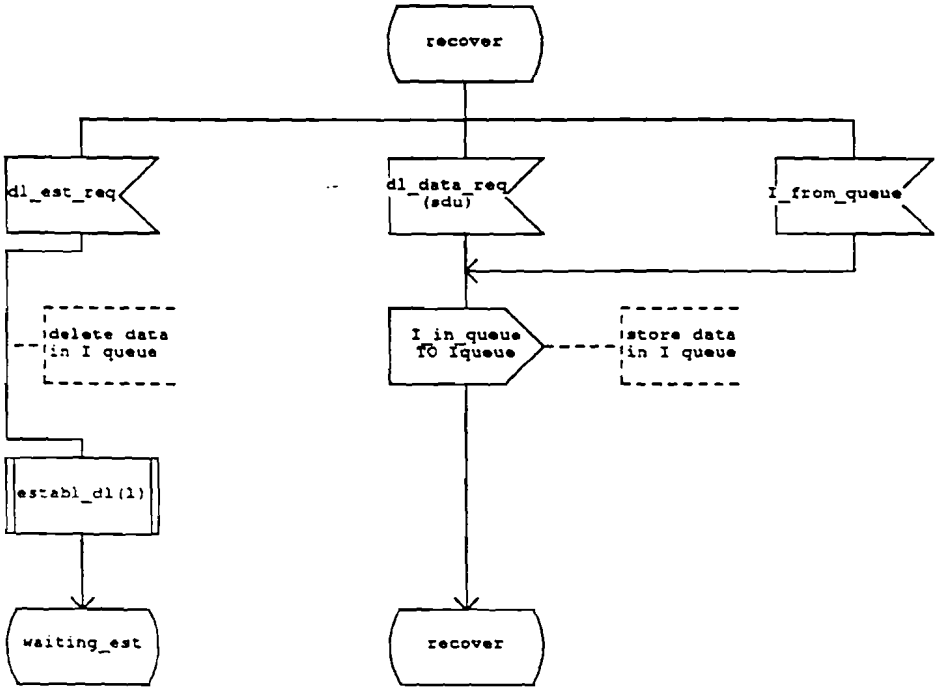
ROCESS LINK_PROCESS 5 (10)



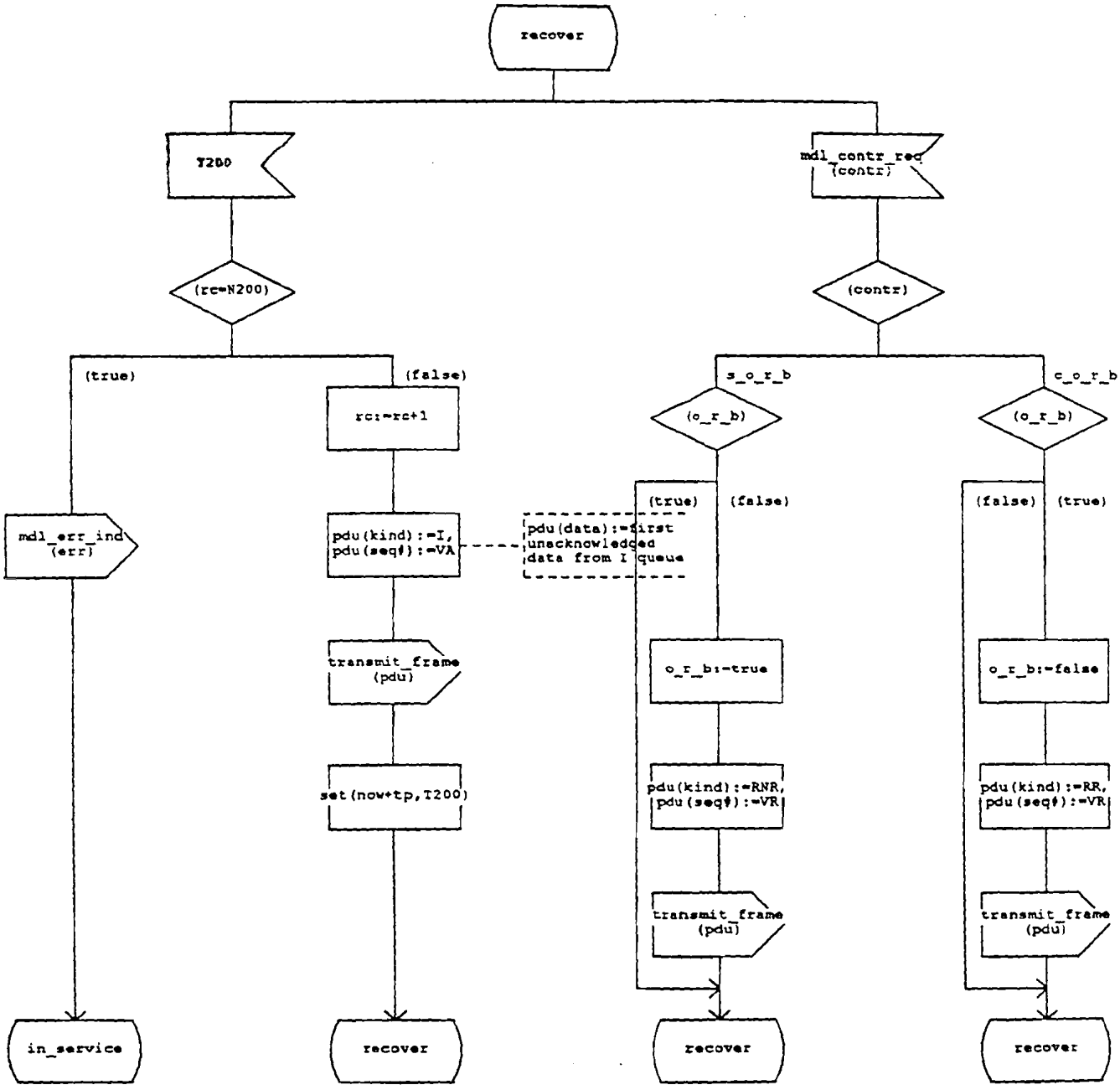
PROCESS LINK_PROCESS 6 (10)



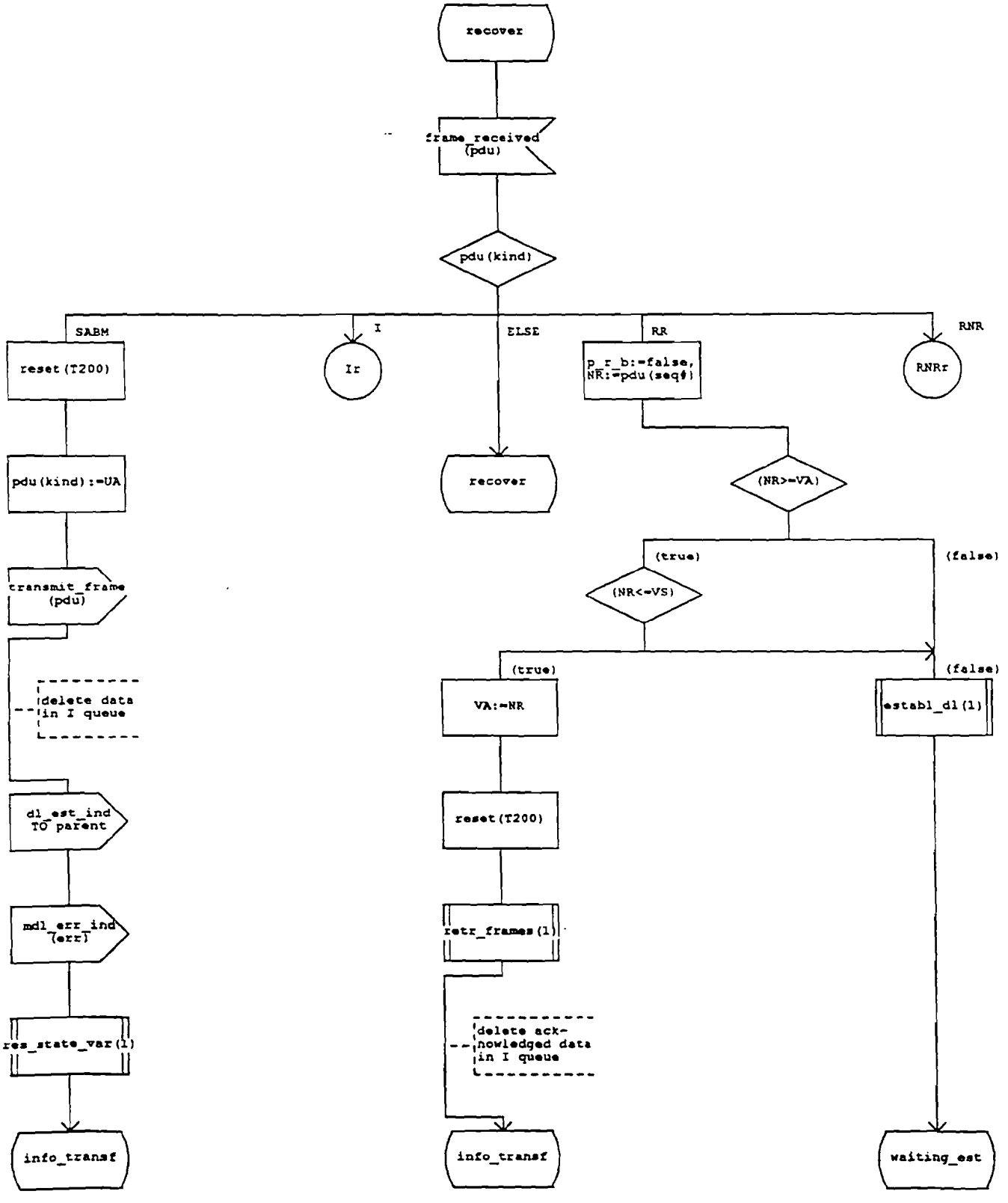
OCESS LINK_PROCESS 7 (10)



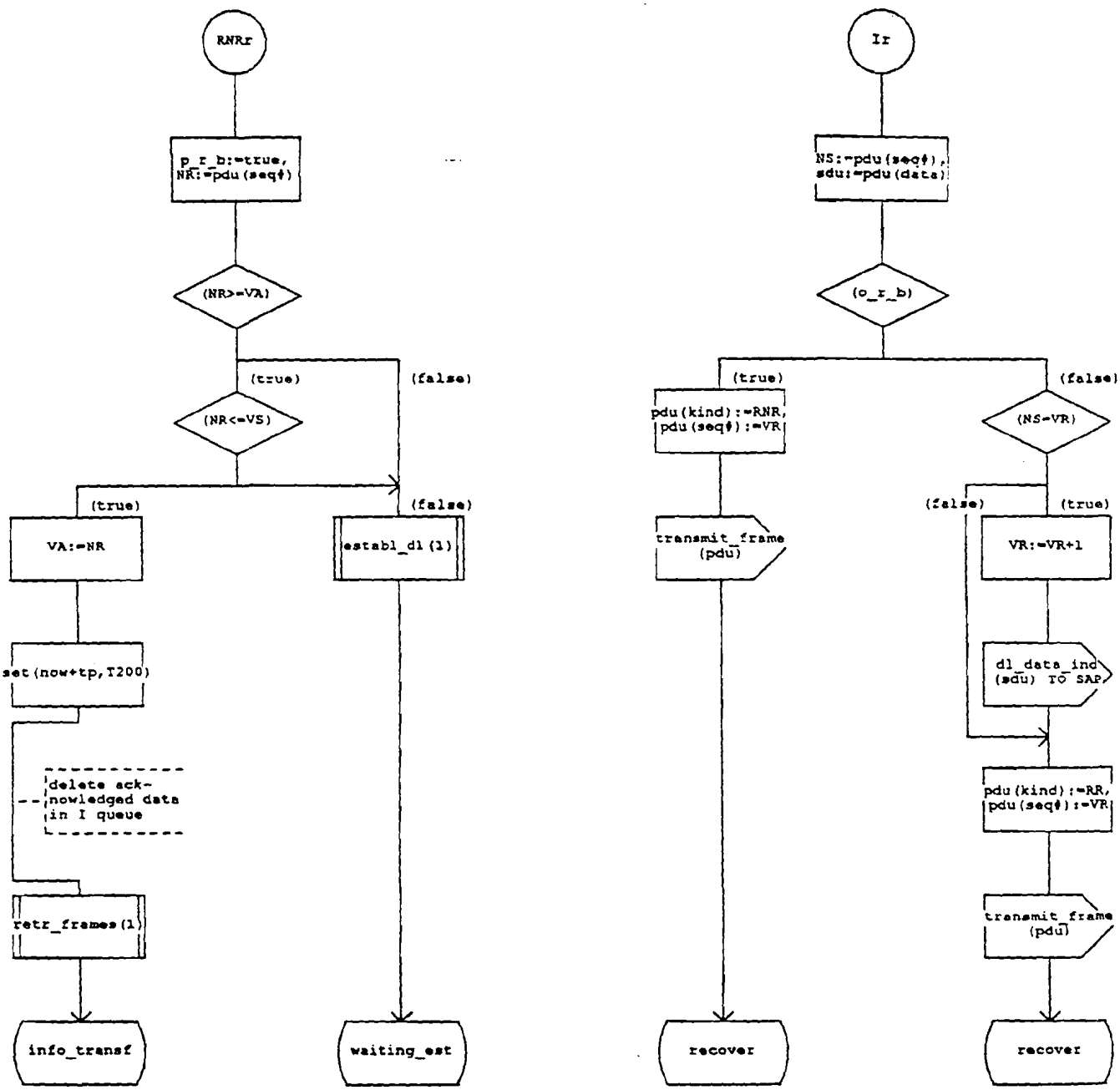
PROCESS LINK_PROCESS 8 (10)



PROCESS LINK_PROCESS 9 (10)



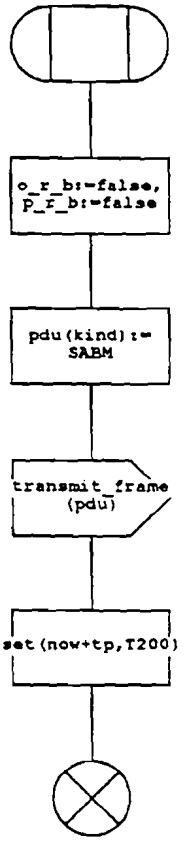
PROCESS LINK_PROCESS 10 (10)



PROCEDURE establ_dl 1 (1)

/* This procedure resets some state variables and tries to establish a data-link */

fpar dummy integer;
/* This is a dummy parameter to overcome a bug in SAC-generate simulator ! */

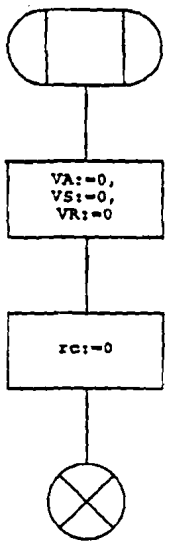


PROCEDURE res_state_var

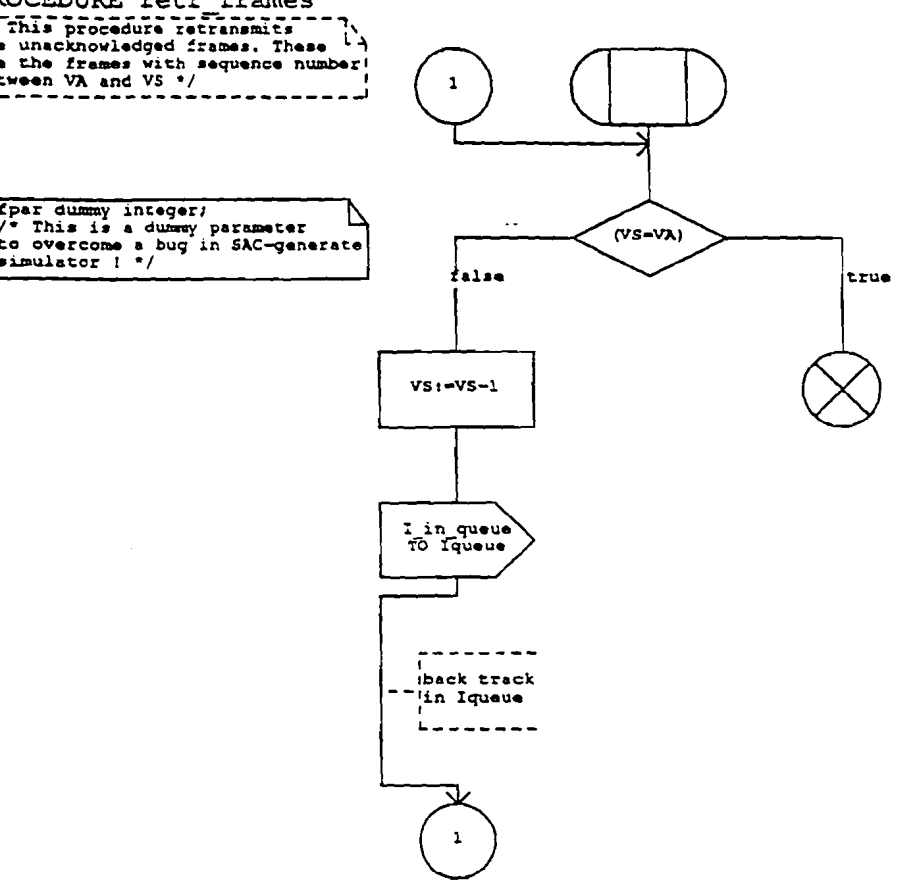
1 (1)

```
/* procedure which resets the
state variabels.
VA : Number of acknowledged frames
VR : Number of next frame to receive
VS : Number of next frame to send */
```

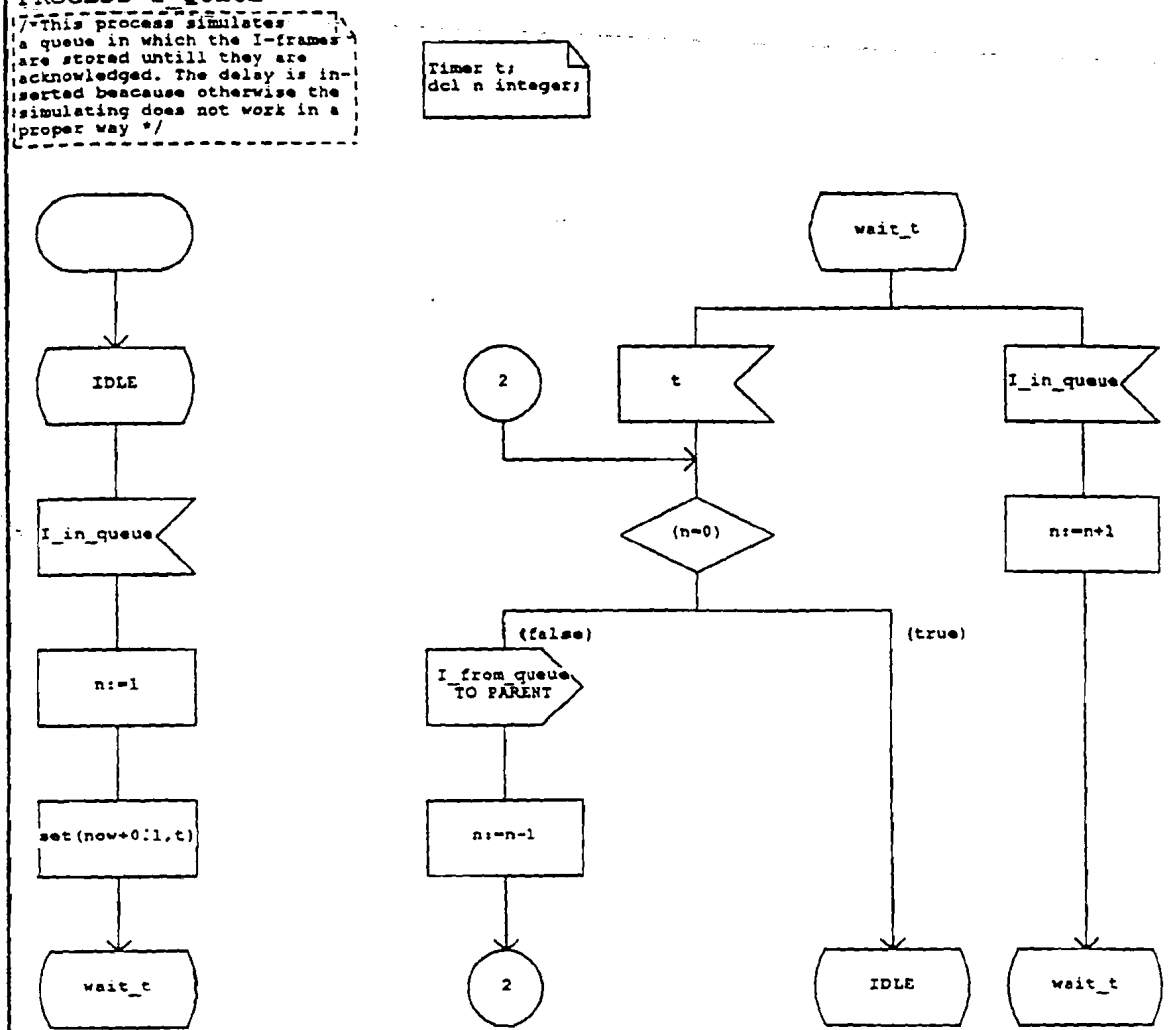
```
fpar dummy integer;
/* This is a dummy parameter
to overcome a bug in SAC-generate
simulator ! */
```



ROCEDURE retr_frames 1 (1)



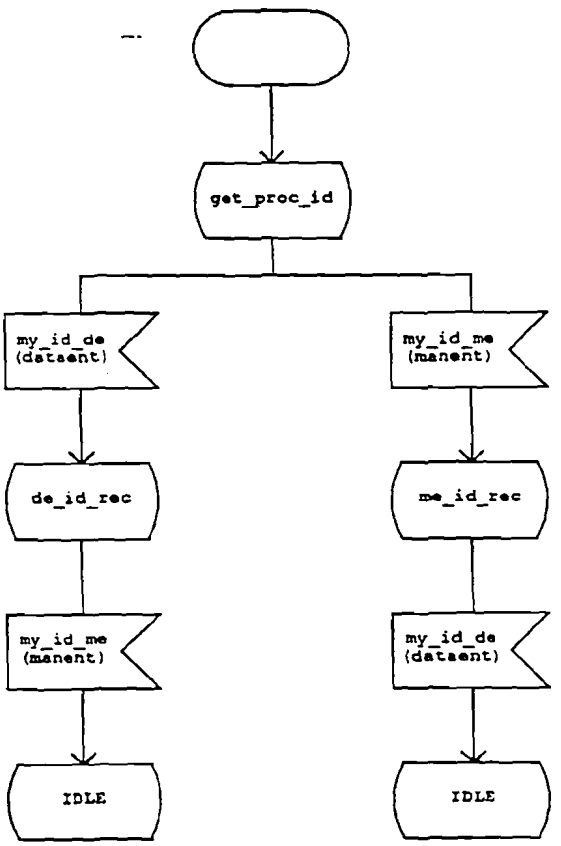
PROCESS I_QUEUE 1 (1)



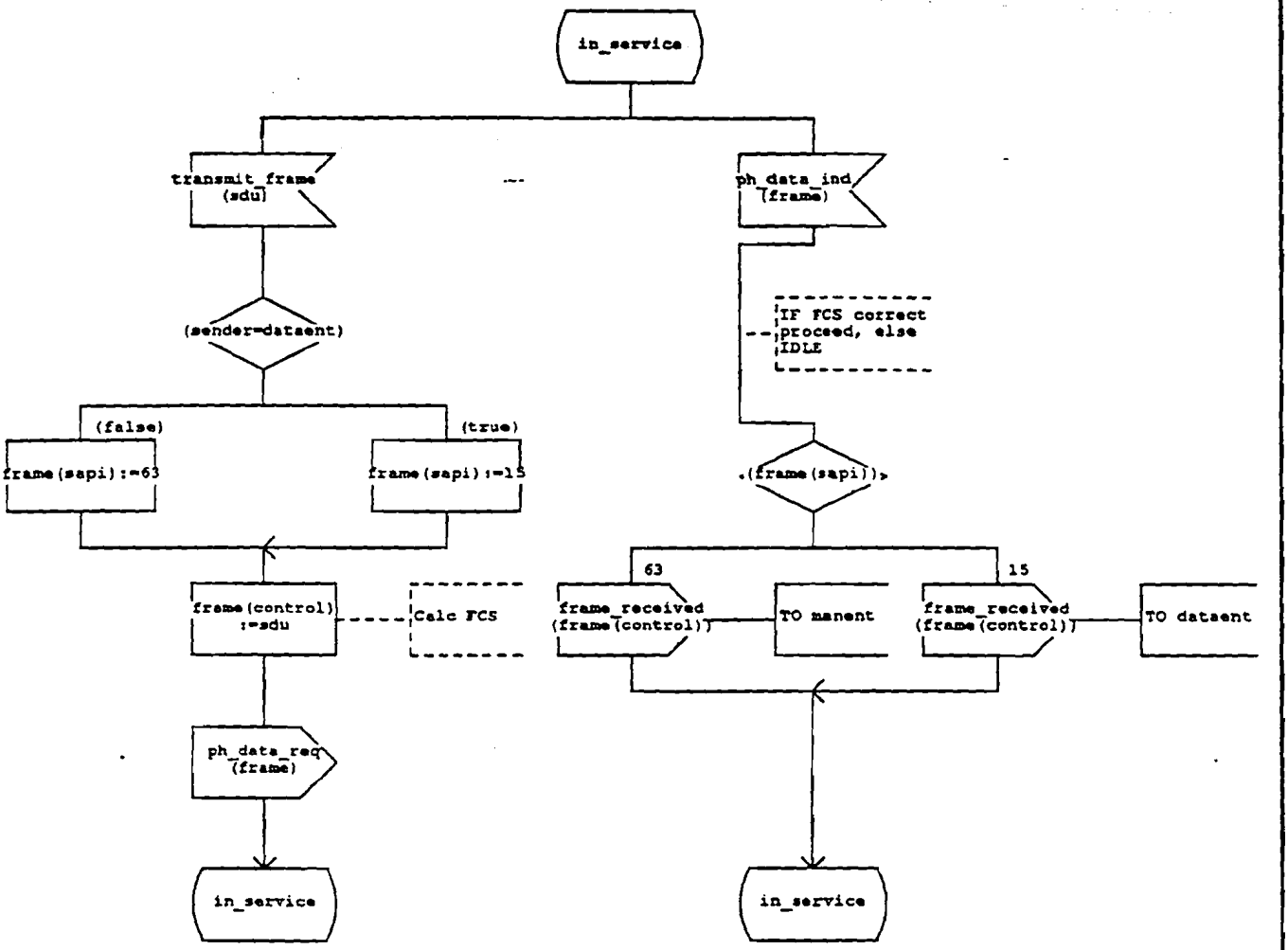
PROCESS MULTIPLEXING 1 (2)

```
/* This process forms the lower
layer of layer 2 (layer 2A). It
multiplexes and demultiplexes the
different datalinks onto/from the
physical layer. It also checks or
generates the frame check sequence */
```

```
dcl dataent,manent PId;
dcl frame frame_type,
sdu control_record_type;
```



PROCESS MULTIPLEXING 2 (2)



# Appendix B

# Simulation of Lapi operation

```
1.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, SABM, FCS .)
1.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 63, SABM, FCS .)
1.00 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, UA, FCS .)
1.00 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 63, UA, FCS .)
1.60 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 0, (. packet .) .), FCS .)
1.60 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 63, (. I, 0, (. packet .) .), FCS .)
1.60 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 1, (. packet .) .), FCS .)
1.60 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 2, (. packet .) .), FCS .)
2.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. RNR, 1 .), FCS .)   /* peer busy for data link*/
3.50 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 63, (. I, 1, (. packet .) .), FCS .)
3.50 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 63, (. I, 2, (. packet .) .), FCS .)
3.50 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 63, (. I, 3, (. packet .) .), FCS .)
3.50 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 63, (. I, 4, (. packet .) .), FCS .)
3.50 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 63, (. I, 5, (. packet .) .), FCS .)
3.50 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 63, (. I, 6, (. packet .) .), FCS .)
3.50 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 63, (. I, 7, (. packet .) .), FCS .)
3.60 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. RR, 3 .), FCS .) /* peer no longer busy */
3.60 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 63, (. RR, 3 .), FCS .)
3.70 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 3, (. packet .) .), FCS .)
3.70 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 4, (. packet .) .), FCS .)
3.70 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 5, (. packet .) .), FCS .)
3.70 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 6, (. packet .) .), FCS .)
3.70 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 7, (. packet .) .), FCS .)
3.70 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 8, (. packet .) .), FCS .)
3.70 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 9, (. packet .) .), FCS .)
3.70 ph_data_req from multiplexing:1   /* last frame, window full */
Parameter(s) : (. 01111110, 16, (. I, 10, (. packet .) .), FCS .)
4.20 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. RR, 4 .), FCS .)
4.20 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 63, (. RR, 4 .), FCS .)
4.30 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 11, (. packet .) .), FCS .)
4.40 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. RR, 5 .), FCS .)
4.40 ph_data_ind to multiplexing:1
```

```
Parameter(s) : (. 01111110, 63, (. RR, 5 .), FCS .)
4.50 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 12, (. packet .) .), FCS .)
4.60 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. RR, 6 .), FCS .)
4.60 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 63, (. RR, 8 .), FCS .)
4.60 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 63, (. RNR, 0 .), FCS .) /* Napi-management busy */
6.60 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 6, (. packet .) .), FCS .) /* timeout */
8.60 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 6, (. packet .) .), FCS .) /* timeout */
10.60 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 6, (. packet .) .), FCS .) /* timeout */
12.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. RR, 8 .), FCS .) /* retransmit from 8 */
12.10 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 8, (. packet .) .), FCS .)
12.10 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 9, (. packet .) .), FCS .)
12.10 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 10, (. packet .) .), FCS .)
12.10 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 11, (. packet .) .), FCS .)
12.10 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 12, (. packet .) .), FCS .)
14.10 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 8, (. packet .) .), FCS .) /* timeout */
16.10 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 8, (. packet .) .), FCS .) /* timeout */
18.10 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 8, (. packet .) .), FCS .) /* timeout */
20.10 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 8, (. packet .) .), FCS .) /* timeout */

*** error to management, more than N200 retransmissions ***

100.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16,  SABM , FCS .) /* establish new data link */
100.00 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16,  UA , FCS .)
102.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 0, (. packet .) .), FCS .)
102.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 63, (. I, 0, (. packet .) .), FCS .)
102.00 data_arrived from Napi-Data:1
Parameter(s) : (. packet
102.00 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. RR, 1 .), FCS .)
102.00 ph_data_req from multiplexing:1      /* Napi-management still busy */
Parameter(s) : (. 01111110, 63, (. RNR, 0 .), FCS .)
103.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 1, (. packet .) .), FCS .)
103.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16,  SABM , FCS .)
103.00 data_arrived from Napi-Data:1
Parameter(s) : (. packet
103.00 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. RR, 2 .), FCS .)
103.00 ph_data_req from multiplexing:1
```

```
Parameter(s) : (. 01111110, 16,  UA , FCS .)
104.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 2, (. packet .) .), FCS .)
104.00 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. RR, 0 .), FCS .)
105.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16,  SABM , FCS .)
105.00 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16,  UA , FCS .)
105.50 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 0, (. packet .) .), FCS .)
105.50 data_arrived from Napi-Data:1
Parameter(s) : (. packet
105.50 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. RR, 1 .), FCS .)
106.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 4, (. packet .) .), FCS .)
106.00 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. RR, 1 .), FCS .)
107.00 ph_data_ind to multiplexing:1
Parameter(s) : (. 01111110, 16, (. I, 1, (. packet .) .), FCS .)
107.00 data_arrived from Napi-Data:1
Parameter(s) : (. packet .)
107.00 ph_data_req from multiplexing:1
Parameter(s) : (. 01111110, 16, (. RR, 2 .), FCS .)
```

# Appendix C

# SDL description of the interworking function

PROCESS IWF 1 (1)