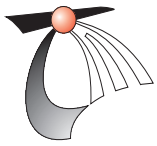
Connectionless Communications using the Asynchronous Transfer Mode



CTIT PH. D-THESIS SERIES NO. 95-03

P.O. BOX 217 - 7500 AE ENSCHEDE - THE NETHERLANDS
TELEPHONE +31-53-893779 / FAX +31-53-333815

CENTRE FOR
TELEMATICS AND
INFORMATION
TECHNOLOGY

Promotiecommissie:

prof. dr. ir. Th.J.A. Popma (voorz.)
prof. dr. ir. J. van Amerongen (secr.)
prof. dr. ir. I.G.M.M. Niemegeers (promotor)
dr. ir. B.R.H.M. Haverkort (referent)
prof. dr. M. El Zarki
prof. dr. P. Martini
prof. dr. ir. F.C. Schoute
dr. H. Ouibrahim
prof. dr. ir. C.A. Vissers
dr. ir. V.F. Nicola

Omslagontwerp: Saskia Marquering

Druk: Universiteit Twente

CIP-GEGEVENS KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Heijenk, Geert

Connectionless communications using the asynchronous transfer mode / Geert Heijenk. - [S.1. : s.n.]. - I11. -
(CTIT Ph. D-thesis series, ISSN 1381-3617 ; no. 95-03)
Proefschrift Universiteit Twente Enschede. - Met lit. opg.
ISBN 90-365-0733-2
Trefw.: telecommunicatie / computernetwerken / asynchrone communicatie.

CONNECTIONLESS COMMUNICATIONS USING THE ASYNCHRONOUS TRANSFER MODE

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. Th.J.A. Popma,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op vrijdag 3 maart 1995 te 16.45 uur

door

Gerhard Jan Heijenk

geboren op 4 augustus 1965

te Gorssel

Dit proefschrift is goedgekeurd door de promotor
prof. dr. ir. I.G.M.M. Niemegeers

Abstract

For a large number of applications, there is a strong need for the provision of a connectionless service by the B-ISDN. However, the cell-based ATM, which is used for switching and multiplexing in this public telecommunication network, is a connection-oriented technique. The B-ISDN should therefore be extended with functionality to allow for connectionless communications. This dissertation addresses the design and analysis of such an extension.

An architectural framework is presented, which places the protocols to be used in perspective. Two possible network architectures result from the functional decomposition of the connectionless service into cooperating protocol entities and the underlying ATM service. In the first one, end-systems of the B-ISDN are interconnected by means of end-to-end ATM connections. In the second one, end-systems are connected to special entities in the B-ISDN, called Connectionless Servers (CLSs). The CLSs are interconnected by ATM connections, thus constituting a connectionless overlay network on top of ATM.

A number of different implementation architectures for a CLS are proposed, and analysed with respect to effectiveness, availability, scalability, and in particular, performance. The major distinction between these implementation architectures is the distribution of functionality over modules. Furthermore, two different modes of operation are identified for a CLS. In the message mode of operation, a packet is reassembled from the incoming cells before it is processed and forwarded. In streaming mode of operation, the first cell of a packet is immediately processed and forwarded upon arrival, while state information is maintained for the processing and forwarding of subsequent cells of the packet.

A number of performance models are developed in this dissertation. An approximate model of a CLS is analysed to allow for comparison of the delay which is experienced by cells for different implementation architectures and modes of operation. If the bandwidth assigned to ATM connections between CLSs is relatively high, message mode of operation yields the lowest delay, otherwise streaming mode performs best.

In order to support the dimensioning of a reassembly buffer in a CLS operating in message mode, another, more detailed model is developed and analysed. It allows the computation of the packet loss probability of a buffer, as a function of its size.

An essential function for the provision of a connectionless service using ATM is connection management. This function instructs the signalling system of the B-ISDN to establish and release ATM connections as needed for the transfer of packets. A new mechanism is proposed that exploits the expected correlation between subsequent packet arrivals to reduce the average bandwidth that needs to be reserved by the ATM network. A performance model is developed and analysed to determine the optimal control parameters of the new mechanism, and to evaluate its behaviour. It is shown that bandwidth reductions of up to 95% can be obtained, compared to conventional mechanisms, without affecting the average delay experienced by packets.

Acknowledgements

When you arrive at the sea, after going down a river from its source to its mouth, it is a good time to reflect on the time you spent paddling. Writing this dissertation was very much like canoeing down a river, and I am now arriving at some kind of delta. Although I had to do most of the paddling myself, I have received help from a lot of people. I would like to thank them all for their contribution to the journey.

First of all, I would like to thank my *promotor* Ignas Niemegeers. His knowledge and understanding have been invaluable to me. He has guided me during the years I worked on the dissertation, giving me an enormous freedom to do what I felt was important and genuine. He has helped me embarking on this trip, and put beacons along the river.

I am very grateful to my *referent* Boudewijn Haverkort, who has put a huge effort in assuring the scientific soundness of this dissertation. The discussions we had, and his suggestions for the presentation of the work have greatly improved the quality of the dissertation. His friendly and cooperative support has guided me to the sea.

In the early years of my journey, I have had the opportunity to sail with a number of people at PTT Research. I would like to thank Frans van den Dool, Harm Mulder, Chris Gabriël, and Rein de Vries for giving me this opportunity, and for being such kind part-time and long-distance colleagues.

Magda El Zarki gave me the opportunity to work at the University of Pennsylvania for a couple of months. It has been very pleasant and fruitful to cooperate with her. I would like to thank her and Pramod Pancha for making my stay in Philadelphia such an unforgettable experience.

A number of students have written their master's theses on subjects related to this dissertation. I would like to thank Paul Swart, Henk-Jan Olde Loohuis, and William van Dieten, who have helped me paddling during parts of the trip.

The members of the TIOS group have created a pleasant and stimulating working environment. I take this opportunity to acknowledge some of them in particular.

Through the years, Martin van der Zee, Xinli Hou, Jan Laarhuis, Ellen ter Brugge, Hans Daemen, Hans Pasch, Marcel Jordense, Frank Baumann, Byung Kim, and Marloes Castañeda have been very friendly colleagues, who have all contributed to my graduation in some manner. Aad van Moorsel has helped me with some of the mathematical problems I encountered. His contributions to this dissertation are greatly appreciated. On the more personal level, I received much in return for shipping his dirty laundry.

I am indebted to Saskia Marquering for advising me in the layout of this dissertation, and for designing the cover. Special thanks goes to Ineke Boersma, Arnoud Kuiper, Rolf Appel, and Stan Megens, and many other friends for minor and major contributions to my trip.

Ingrid Schulten has accompanied me on large parts of the journey. Her thoughtful sympathy will never cease to warm my heart.

Finally, I would like to say ‘dankjewel’ to my parents, Jan and Diny Heijenk, who taught me the value of personal development. They were the source of the river.

Contents

If you think technique is meaning,
you might find me very simple.

Lou Reed & John Cale - Songs for Drella

Chapter 1
Introduction

This dissertation is about communications. Communication between human beings is considered essential to our society ([45]). Currently, communication between non-humans, in particular between machines, is becoming increasingly important. Furthermore, machines can be extremely useful to support the communication between humans that are separated by time or place.

Although we believe that the importance of inter-human communication is of a higher order than that of inter-machine communication, we will focus on the latter sort of communication. This dissertation is concerned with the design and analysis of a system, i.e., a set of cooperating machines, which provides other machines and indirectly humans with certain communication capabilities. We will draw some parallels with inter-human communication to illustrate the particular area of inter-machine communication that is the subject of this dissertation.

Basically two types of communication can be identified. These are referred to as connectionless communication and connection-oriented communication. Communication between humans can be considered to be *connectionless* if an amount of information is passed from a party to one or more other parties as a self-contained unit, as a message. The parties do not need to engage in an association, such as a conversation, before the communication can start. There is no agreement on the communication prior to the offering of the self-contained unit of information. You are now reading this dissertation (and probably still wondering what it is about). I wrote this dissertation some time ago. For this communication from me to you, we did not have to make an appointment. I have just arranged that you received this dissertation. For the rest, this dissertation is a self-contained unit of information. You have probably already read that I am the author, i.e., the source of the information. This dissertation contains an introductory and a concluding chapter. It contains all I intend to communicate to you regarding “Connectionless Communications using the Asynchronous Transfer Mode” (and for many of you much more).

Communication between humans can be considered to be *connection-oriented* if there is an association between the parties, on which all parties agreed at the beginning of the communication. This association forms a context, describing aspects of the communication, e.g., the communicating parties. During the existence of the association, the parties exchange information as required. Probably, you are left with some questions after reading this dissertation. If you contact me and ask: “Can I ask you a question about your dissertation?”, and I answer positively, we are initiating a connection-oriented communication.

Inter-machine communication is connection-oriented if a logical association (a connection) between the communicating machines is explicitly established at the beginning of the communication, and released at the end. During the existence of the connection, all machines are involved in the communication. Inter-machine communication is connectionless if a machine constructs a piece of information (a packet), addresses it explicitly to one or more other machines, and sends it. At the moment one of the destination machines receives the packet, the sending machine may very well be busy with other tasks. The communicating machines do not necessarily have to be involved in the communication all at the same time.

This dissertation is about connectionless communications. It deals with the problem how to support the increasing demand for this type of communication with modern *telecommunication networks*, i.e., with systems of cooperating machines that support communications over a wide geographical span. The demand for communications, connection-oriented as well as connectionless, is increasing rapidly. Not only the number of users that want to attach to communication networks with their terminals, i.e., with machines such as telephones or computers, is increasing. The amount of information that must be transferred on behalf of the attached users is increasing even more rapidly. The term bandwidth is used to denote the amount of information that is transferred by a communication network per unit of time. Broadband communications refers to communication with a high bandwidth, typically in the order of millions of bits (Mbits) per second. The purpose of this dissertation is to give insight in some important aspects of the design and analysis of systems that can support broadband connectionless communications.

This dissertation is about connectionless communications using the Asynchronous Transfer Mode (ATM). ATM is a switching and multiplexing technique, which has been designed to support broadband communications in modern telecommunication networks ([46]). Furthermore, ATM is considered to be very suitable to support different types of communications, such as telephony, TV

distribution, and data communication, integrated within a single network. Traditionally, these different types of communication have been provided by separate networks, e.g., telephony networks, CATV networks, and data networks. ATM is believed to be a technique that is capable of supporting all different types of communication with a single integrated broadband communication network.

ATM has been developed as a connection-oriented technique. Therefore extensions should be made to the telecommunications network for the support of connectionless communications. The collection of ATM and the mentioned extensions constitutes a system for broadband connectionless communications. The design of such systems is what this dissertation is about.

We aim at a system that provides for broadband connectionless communications over a wide-area public telecommunications network. The Broadband Integrated Services Digital Network (B-ISDN), which will use ATM as the underlying technique, is envisaged to be the future public telecommunications network for the wide area ([16], [55]). Therefore, the work described here is performed in the context of B-ISDN.

The work aims at designing a general connectionless service, which can be used by all types of applications. We aim at the most general system that provides end-to-end connectivity for connectionless communications. In the well known Reference Model for Open Systems Interconnection (OSI-RM) ([65]), this can be compared to the service of the Network Layer. Furthermore, the purpose of the work is to use ATM as it has been defined within the International Telecommunication Union (ITU) ([73]), and to add functionality to the B-ISDN in order to provide a connectionless service. Since ATM is sometimes considered to provide a rich Physical Layer service ([111]), the work of this dissertation can roughly be positioned at the Datalink and Network Layer of the OSI-RM.

Most problems that are encountered during the design stem from the difference between the required characteristics of the service to be provided and the characteristics of the underlying ATM service. The service to be designed is connectionless, while ATM is connection-oriented. The connectionless service should allow for the transfer of packets of any size, whereas ATM has been standardized to transfer small fixed-size packets. Functions that are essential to bridge this difference in nature are the following. The *segmentation and reassembly* function transforms the variable size packets to the fixed-size ATM packets, and vice versa, by cutting packets into chunks, and by collecting the chunks to retrieve the original packet again. The *connection management* function makes sure that a connection

for the transfer of a (connectionless) packet is available when needed, by establishing and releasing ATM connections.

The work presented in this dissertation does not cover the entire design trajectory, from user needs to realization (see e.g., Figure 2 in [138] for an overview of the design trajectory). We start with user requirements in the literature, and in the specifications given by standardization bodies such as the ITU. After performing a number of design steps, we propose a network architecture. This architecture identifies different functional entities, their interaction, and the protocols according to which they cooperate.

Some of these entities will be available in an ATM based B-ISDN, i.e., they will be needed to provide the ATM service. Others have to be specially designed. In the public part of the network, these entities will be realized by means of so-called Connectionless Servers (CLSs). As a first step into the design of a CLS, we present a number of implementation architectures, which identify different modules, the functions they perform, and the way they interact. A number of important issues in the implementation of a CLS are further explored.

During the design process, we analyse the design (or the design alternatives) with respect to their effectiveness, i.e., we check if they exhibit the required functional behaviour. Additional design criteria that are considered are performance, availability, and scalability. Availability refers to the ability of a system to continue operation, even when certain system components fail. Scalability refers to the possibilities to extend the system so that it can accommodate a higher load.

The analysis of the performance of the system under design aims at obtaining insight in measures that are of importance for the users of such a system, e.g., delay, throughput, and loss probability. Furthermore, it is often considered interesting to be able to quantify the resources that are needed to obtain a certain performance, e.g., for dimensioning purposes. Performance analysis constitutes an important part of the work presented in this dissertation.

This dissertation consists of four parts. The first part, which is the introductory part, contains the current chapter and Chapter 2. The latter introduces concepts that are essential to the dissertation, such as ATM and B-ISDN, and expands on the notion of a connectionless service. Furthermore, it explores the applications the connectionless service is being designed for, and surveys related work in the literature.

The second part, containing the chapters 3, and 4, describes the design and functional analysis of the connectionless service. Chapter 3 presents the design of the network architecture, starting from the requirements. Chapter 4 is concerned with the design of a CLS. It presents several alternative implementation architectures, and analyses them with respect to their effectiveness, availability, and scalability.

The third part, containing the chapters 5, 6, and 7, presents the performance studies that are done to analyse the design of the second part. Chapter 5 describes the analysis of a CLS. In the other two chapters, we analyse two functions, which have already been mentioned as essential for the design, in more detail. Chapter 6 describes the performance analysis of the reassembly function. Chapter 7 describes the analysis of the connection management function.

The fourth and last part of this dissertation, Chapter 8, summarizes the results of the work, and draws conclusions.

And I believe
the aliens have to take a physical form on our planet.
So why not one with 13 channels?

Joe Jackson - Night and Day

Chapter 2
Concepts, Applications, and Related Work

A lot of effort is currently being invested in the design of broadband communication systems. The design of a system for supporting connectionless communications should be seen as an integral part of these activities. In this chapter we review some basic concepts in B-ISDN, discuss applications that will use a connectionless service, and survey related work. The purpose of this chapter is to introduce the information that forms a starting point for the work presented in the rest of the dissertation.

The organization of this chapter is as follows. First, in Section 2.1, a number of concepts that are relevant to the dissertation are introduced. In Section 2.2 we look at applications that may want to use connectionless communication. In Section 2.3 we survey related work. We conclude this chapter by summarizing it, and giving concluding remarks, in Section 2.4.

2.1 Basic Concepts

The concept of connectionless communication is essential to this work, and can only be understood in relation to its opposite, i.e., connection-oriented communication. Both are discussed in Section 2.1.1. The network we are considering for providing connectionless data communication is the B-ISDN, which will be introduced in Section 2.1.2. Finally, in Section 2.1.3, we will briefly review some basic concepts of ATM, which is the network technology that will be used to support connectionless data communication.

2.1.1 Connectionless versus Connection-oriented

In this dissertation the terms connectionless and connection-oriented are used as a characteristic of a service, i.e., the observable behaviour of a communication system. A service is *connectionless* if two or more users of the service can transfer data using the communication system *without* first establishing and later releasing a connection. A service is *connection-oriented* if users *must* establish a

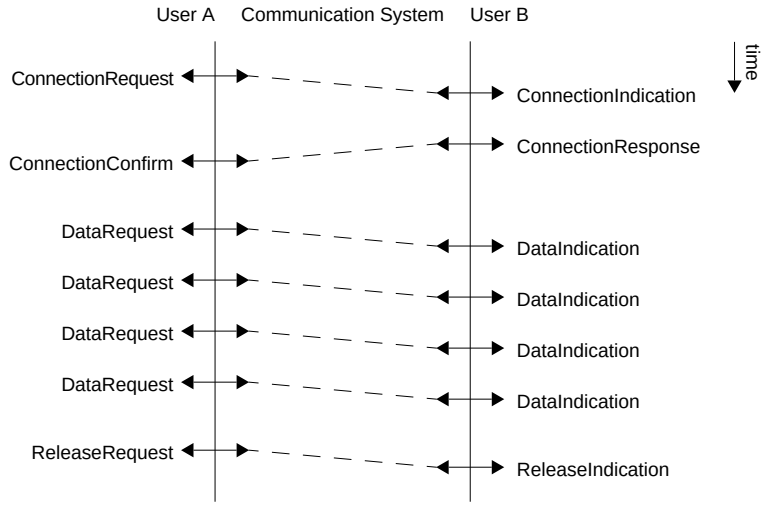


Figure 2-1: Sequence of Service Primitives for a Connection-oriented Service

connection before they can transfer data using the communication system. The establishment of a connection is a negotiation between the users who wish to communicate and the communication system. During the negotiation, state information related to the connection is exchanged between the parties. The communication system reserves resources for a connection, e.g., bandwidth.

The essential interactions between a communication system and its users can be described by means of the sequence of service primitives, which are exchanged at the boundaries between the system and its users. Figure 2-1 shows a likely sequence of service primitives in case of a connection-oriented service, where two users want to communicate. Service primitives are denoted by double arrows, to indicate an interaction in which both the user and the communication system are involved. First, the initiating users requests the system to establish a connection (ConnectionRequest). In this request at least the address of the required destination, and often parameters regarding the characteristics of the traffic and the required Quality of Service (QoS) are passed to the system. The system analyses the parameters of the request and determines to what extent it can support the requested connection. If the connection can be supported, a ConnectionIndication primitive occurs between the system and the called user. This user checks whether and under what conditions it wants to accept the connection. The acceptance of the connection is indicated to the system by another interaction (ConnectionResponse). Finally the initiating user is informed of the successful establishment of the connection (ConnectionConfirm). From now on, the users can transfer data. The data is passed from the sending user to the system by means of a DataRequest primitive, transported by the system, and passed from

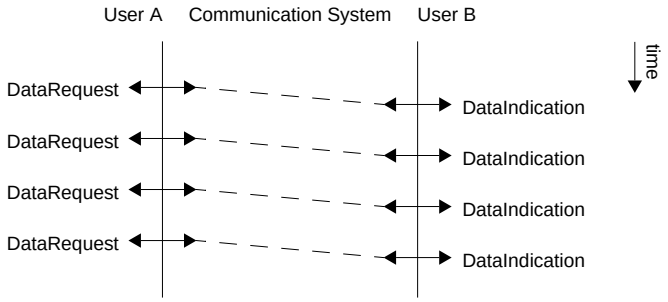


Figure 2-2: Sequence of Service Primitives for a Connectionless Service

the system to the receiving user by means of a DataIndication primitive. After all data has been transferred, the connection should be closed again. Therefore, one of the users, not necessarily the initiating user, informs the system that it wants to release the connection (ReleaseRequest). The system releases the connection and informs the other user (ReleaseIndication).

For a connectionless service, the sequence of service primitives is much simpler (Figure 2-2). No primitives for establishing or releasing a connection are needed. Each data unit is transferred individually. The sending user passes the data to the system by means of a DataRequest primitive. The system transports the data and passes it to the receiving user by means of a DataIndication primitive. All information regarding the transfer of the data unit, e.g., required destination address, should be passed between the users and the network in these primitives.

An essential characteristic of a connectionless service is that the system does not have any advance knowledge about the data that should be transferred (e.g., destination, or rate at which data units are offered to the system). In case of a connection-oriented service this information is agreed upon during the connection establishment, so that the system can reserve resources for the transfer of the data units.

For both types of service there are classes of applications that they support best. Applications that require a direct association between the users, such as telephony, are best served by a connection-oriented service. Other applications, such as those which involve the transfer of only a single unit of information between a source and a destination are better served by a connectionless service. Moreover, there are a huge number of applications that use the TCP/IP protocol suite. These applications require a connectionless service at the network layer, also when they will be supported by the B-ISDN.

It is possible to use different types of services at different layers in the network. In [82] the use of connectionless protocols at the network layer, to support all types of applications is advocated. Others prefer to use a single connection-oriented protocol, i.e., ATM, to support all applications. Since ATM has been standardized for future integrated networks, we assume ATM as the basic method to serve all applications. However, we believe that additional provisions have to be made in the network to accommodate applications that are connectionless in nature.

2.1.2 B-ISDN

The Broadband Integrated Services Digital Network (B-ISDN) is expected to be the major future telecommunications network for the wide area. It will provide a wider range of possible applications, and support much higher throughputs than present telecommunication networks. Applications, which can be very diverse in nature, will all be supported by a single network. Moreover, a user can access the network via a single standardized interface, the User-Network Interface (UNI).

The applications that should be supported by the B-ISDN are not only very diverse; they also put very diverse requirements on the network. The network should support:

- point-to-point as well as multi-point communications;
- single-medium as well as multimedia communications;
- connectionless as well as a connection-oriented communications;
- narrowband as well as broadband communications;
- communications involving constant bit rate as well as variable bit rate traffic; and
- communications for applications with a very diverse range of QoS requirements.

Some examples of applications that are foreseen for the B-ISDN are video-conferencing, High Definition Television (HDTV) distribution, video-on-demand, telephony, and applications that are currently supported by the Internet.

In order to support all these applications, the Asynchronous Transfer Mode (ATM) has been adopted for multiplexing and switching in the network ([16]). ATM is a technique, where all user information is transferred in Protocol Data Units (PDUs) of fixed size, called cells. Cells are identified by means of a header,

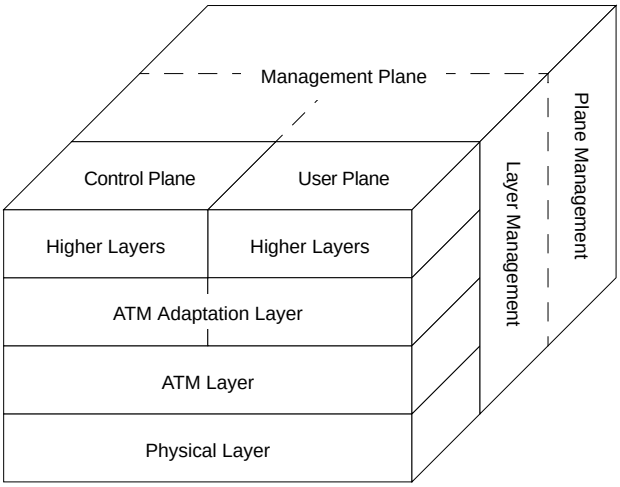


Figure 2-3: The B-ISDN Protocol Reference Model

so that they can be identified for (de)multiplexing and switching. ATM should also be used at the UNI of the B-ISDN. Two different UNIs have been defined ([78]); one with a bit rate of 155.520 Mbits/s, and one with a bit rate of 622.080 Mbits/s.

A Protocol Reference Model (PRM) has been defined for the B-ISDN in [18] (see Figure 2-3). The PRM is divided into planes and layers. The division into (vertical) planes is done to identify different types of functionality, i.e., for the transfer of user information, for the control of calls and connections, and for management. The division into (horizontal) layers is done to create a stepwise independency between the medium, used for the transmission of signals, and the applications. A layer uses the next lower layer to provide a certain, less medium dependent, and more application dependent, service to the next higher layer.

The B-ISDN PRM distinguishes between three planes. The user plane contains the functions that deal with the user information. The control plane contains functions for the control of calls and connections, and the transport of information on behalf of these functions (signalling). Finally, the management plane provides for the coordination between user and control plane, and for the management of individual entities and the overall network. The management plane has been subdivided into layer management, performing management functions for specific protocol layers and their entities, and plane management, performing coordination between the planes and management of the system as a whole.

Within the user and control plane, protocol layers are identified. The lower layers are common to both planes. The Physical Layer provides for the convergence of

ATM cells to signals that can be transferred over a physical medium. The ATM Layer provides for the end-to-end transfer of cells along a connection. It performs switching and multiplexing of cells from different connections. The ATM Adaptation Layer (AAL) adapts the common service provided by the ATM Layer to a service that can better support specific classes of applications. It provides for instance for segmentation and reassembly and for synchronization of source and destination.

Different views exist for relating the layering adopted in the B-ISDN PRM and the layering of the Reference Model for Open Systems Interconnection (OSI-RM) ([65], [110], [111]). In the context of this dissertation, it is convenient to situate the AAL in the OSI Datalink Layer. The ATM Layer can be considered as an upper sublayer of the OSI Physical Layer, or as a lower sublayer of the Datalink Layer.

The applications to be supported by the B-ISDN have very diverse communication requirements. The ATM Layer is common to all applications. The purpose of the AAL is to adapt the ATM service to the service required for the specific applications requirements. In order to come up with a limited number of AAL services, applications have been classified according to the following communication requirements:

- timing relation between source and destination (required or not required);
- bit rate (constant or variable); and
- connection mode (connection-oriented or connectionless).

Since not all combination of the above requirements are foreseen, only four service classes are distinguished according to Table 2-1. Clearly, the class of AAL service we are interested in for the support of connectionless data communications over ATM is service class D. This class does not provide a timing relation between source and destination, supports variable bit rate traffic, and is connectionless.

	Class A	Class B	Class C	Class D
Timing relation between source and destination	Required		Not Required	
Bit Rate	Constant	Variable		
Connection Mode	Connection-oriented			Connectionless

Table 2-1: AAL Service Classes

For each of the service classes, one or more protocols have been defined that can provide the service. For class D, two protocols, called AAL 3/4 and AAL 5 have been defined ([20], [75])[1]. These protocols will be elaborated on in Section 3.4.

For further reading on B-ISDN we refer to [46], [106], or [110].

2.1.3 ATM

The Asynchronous Transfer Mode (ATM) is a technique that is used for switching and multiplexing in the B-ISDN. It can route fixed-size data units, called cells, through a network of switches interconnected by links, from source to destination. ATM is connection-oriented, i.e., prior to the transfer of cells, a connection is established through the network, and cells are forwarded along the connection subsequently.

Multiplexing of cells from different connections on an ATM link is done using asynchronous time division. Time is divided into slots, in which a single ATM cell fits. Slots can be assigned to connections asynchronously, i.e., whenever a slot is needed for a connection it can be used. Identification of the connection a cell belongs to is done by means of a label in the header of the cell. Figure 2-4 gives an example of a series of cells, which are transmitted on a link consecutively. All cells with the same label, e.g., 7, are identified as belonging to a certain connection.

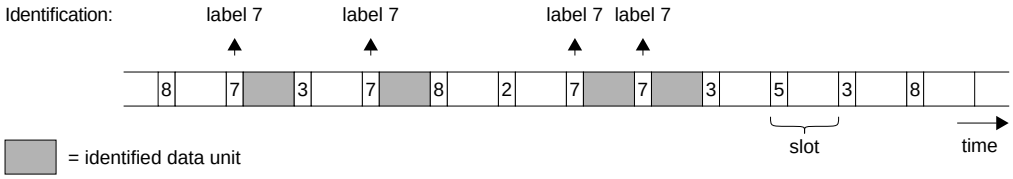


Figure 2-4: Asynchronous Time Division Multiplexing

The alternative to asynchronous time division multiplexing is synchronous time division multiplexing, which is used in present telecommunication networks. Time is again divided into slots, and slots are grouped into frames. Slots are assigned to a connection synchronously, i.e., a connection is assigned the same number of slots in each frame. Identification of the connection a slot belongs to is done by means of the position of a slot in the frame. Figure 2-5 gives an example

[1] Although AAL 3/4 and AAL 5 are intended to support connectionless services, the protocols themselves are connection-oriented. A connectionless protocol should be used on top of the AAL protocol to adapt to the connectionless nature of the applications (see Chapter 3).

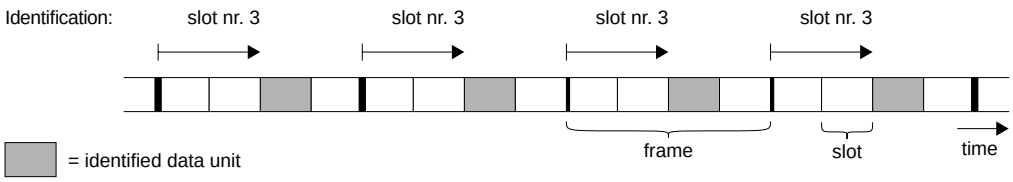


Figure 2-5: Synchronous Time Division Multiplexing

of a series of consecutive cells in this case. All cells which are the third one of a frame are identified as belonging to a certain connection.

An important advantage of asynchronous over synchronous time division multiplexing is that the transmission capacity assigned to a connection can vary in time, depending on the needs of the application. Furthermore, unlike with synchronous time division multiplexing, where the assigned capacity is an integer multiple of the smallest possible capacity (one slot assigned per frame), any capacity can be assigned to a connection.

ATM switches receive cells on a number of incoming links. The connection an incoming cell belongs to is uniquely defined by the incoming link, and the label carried in the cell header. In the switch a routing table is maintained, in which the outgoing ATM link, and the required label for the connection on that ATM link is given for each connection. Using this table, the switch can replace the label of a cell, and forward it to the proper outgoing link. Figure 2-6 gives an example of ATM switching. The shaded entry in the routing table indicates that all cells that arrive on link 2 with label 7 should be forwarded to link 4, while the label should be modified to 3. ATM cells are drawn as in Figure 2-4, i.e., from left to right, first the header, which contains the label, and then the payload.

As stated before, ATM is a connection-oriented technique. The header of an ATM cell relates the cell to a previously established ATM connection. Two types of ATM connections are identified, Virtual Path Connections (VPCs) and Virtual Channel Connections (VCCs). Therefore, also two identifiers can be found in the header of the cell, a Virtual Path Identifier (VPI) and a Virtual Channel Identifier (VCI). The combination of the two, referred to as VPI/VCI, determines the ATM connection a cell belongs to.

A physical link between two switches carries a number of Virtual Path Links (VPLs), each of them is identified by a VPI. The concatenation of a number of VPLs forms a Virtual Path Connection (VPC). Within a VPC, a number of Virtual Channel Links (VCLs) can be identified. Each VCL is identified by a VCI, which is

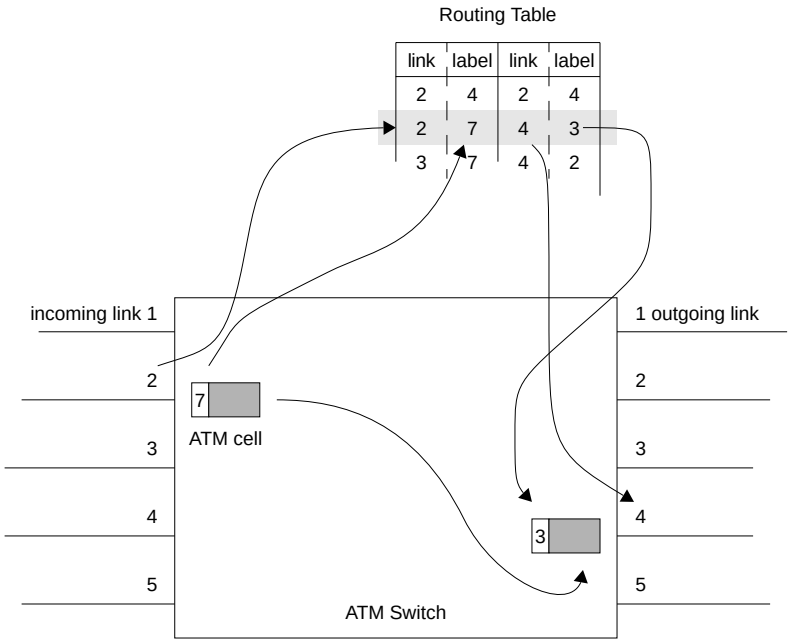


Figure 2-6: ATM Switching

unique for that VPC. The concatenation of a number of VCLs forms a Virtual Channel Connection.

Figure 2-7 shows the relationship between the different types of links and connections. In the figure, a series of 7 ATM switches is shown. On a link between a pair of switches, a number of VP links can be identified. A concatenation of VP links forms a VP connection, e.g., from the first to the third switch. All switches operate on VPs; only some of them operate also on VCs. These are also visible at the VC level. Within a VP connection (between two VC switches), a number of VC links can be identified. The concatenation of a number of these links forms a VC connection, e.g., between the first and the seventh switch.

An ATM cell has a payload field of 48 octets. The header of the cell is 5 octets long. The major fields in the header are the VPI and VCI field. Furthermore, the header contains a Header Error Control (HEC) field to protect the header against bit errors. A Cell Loss Priority (CLP) field can be used to indicate the relative importance of the cell in the connection. The Payload Type (PT) field carries additional information, e.g., an ATM-user-to-ATM-user indication that is transported transparently by the network. At the UNI, also a Generic Flow Control (GFC) field is present in the header in order to control the access to a shared link.

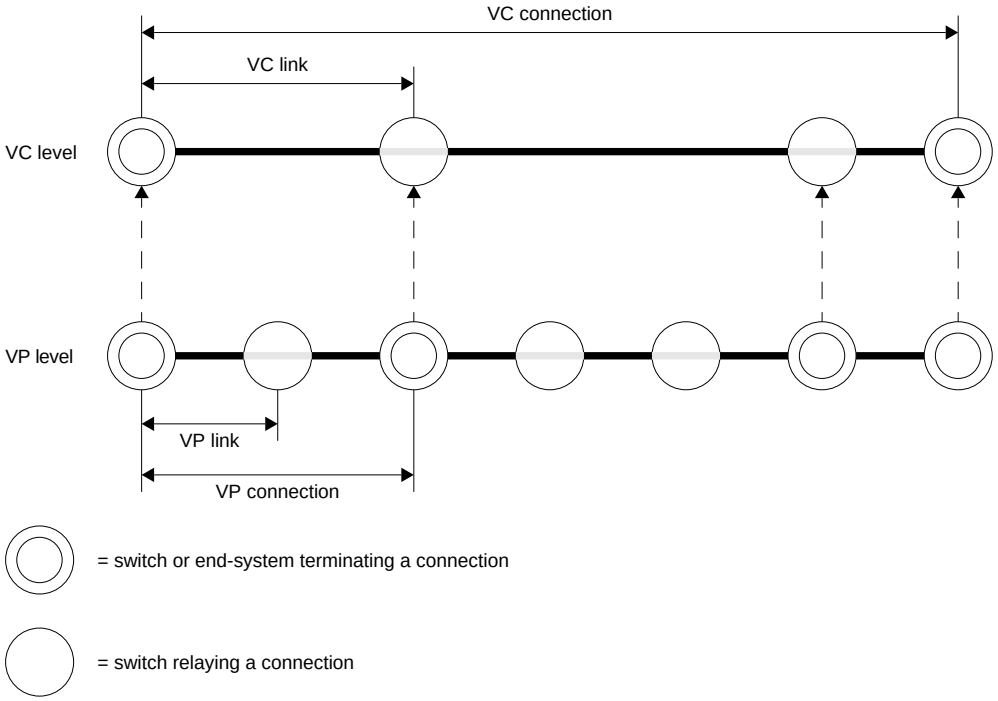


Figure 2-7: ATM Connections

Error control should be performed in layers above ATM on an end-to-end basis. Only errors in the header are detected or corrected (depending on the mode of operation) by means of the 8 bit HEC field, if possible.

In order to protect an ATM network against overload and guarantee the requested QoS to the users, bandwidth is reserved for each connection. A Connection Admission Control (CAC) mechanism checks whether or not bandwidth for a newly requested connection is available. If not, the connection is refused. If the bandwidth is available, the connection is established, and the bandwidth is reserved. In order to check if the user does not violate the agreed bandwidth another mechanism, called Usage Parameter Control (UPC), has to be implemented at the border of (the public part of) the network, directly after the UNI. Cells that cause a violation of the agreed bandwidth are either discarded directly, or given a low cell loss priority, so that they are the first ones to be discarded in case of congestion.

For further reading on ATM we refer to [46], [106], or [110].

2.2 Applications using Connectionless Communications

The emphasis in this dissertation is on connectionless communications over the wide area, with throughputs that have not been available up to now. In order to provide insight into the requirements on the system, we will explore the characteristics of applications that have been identified in a number of publications, e.g., [5], [39], [95], [96], [106], [128], [146]:

- file transfer,
- terminal access,
- information retrieval (e.g., World Wide Web),
- computer graphics,
- distributed supercomputing,
- remote procedure call (RPC),
- virtual memory page swapping and paging, and
- electronic mail.

Although not all applications listed here need necessarily be supported by a connectionless network, most of them are connectionless in nature. This means that the applications need to transfer one or more individual pieces of information, not a stream of related pieces of information. Electronic mail is an example of an application that is inherently connectionless. Its purpose is to transfer individual messages from source to destination. There is no need for an association between source and destination before the message is transferred.

The end-systems that implement the applications can be attached directly to the B-ISDN. However, most applications have been first introduced in a local environment, so that a lot of existing end-systems have been attached to Local Area Networks (LANs). Another reason for attaching end-systems to LANs is the locality of a large share of the generated traffic.

However, there is a growing demand for communications over a wider area. This can be accommodated by interconnecting LANs by means of networks that span a wider geographical area. Most LANs use connectionless protocols. In order to avoid complicated and costly protocol conversion, it is often desirable to interconnect these LANs by means of connectionless networks. A first stage into this

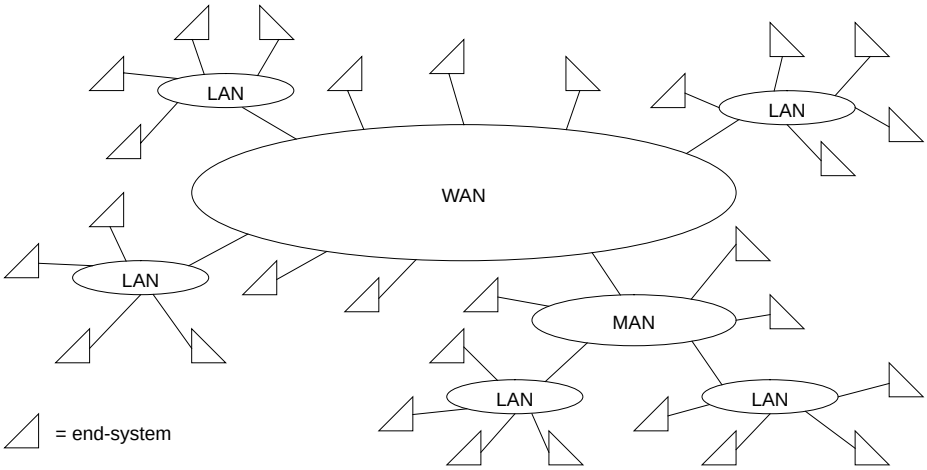


Figure 2-8: Connection of End-systems, LANs, and MANs to a WAN

development is the interconnection of LANs by Metropolitan Area Networks (MANs), which typically have a geographic span of about 50 kilometres. Many of these MANs are based on the Distributed Queue Dual Bus (DQDB) technique, defined in the IEEE 802.6 standard ([63]).

If a Wide Area Network (WAN) supporting broadband connectionless communications becomes available, it can be used to interconnect these MANs, and to interconnect LANs directly. Furthermore, end-systems with high communication needs can be attached directly to the WAN. This will lead to a scenario where end-systems are attached to either a LAN, a MAN, or a WAN, with LANs being connected to either a MAN or a WAN, and the MANs connected to the WAN (see Figure 2-8).

We refer to the systems that interconnect different networks, and perform the needed conversion, as Interworking Units (IWUs). These IWUs can function as routers, interconnecting the different networks at the network layer, e.g., using the Internet Protocol (IP) ([107]) or the ISO Connectionless Network Protocol (CLNP) ([66]). The IWU can also function as a bridge, interconnecting the different networks at the medium access control (MAC) sublayer ([128]).

It is expected that most traffic to be transported by the wide area connectionless network will initially come from end-systems attached to LANs. Interconnection of these LAN is often referred to as the most important application of the network. In fact, the real applications to be supported are those listed above. From the point of view of the WAN, the LAN only aggregates the traffic generated by these applications.

In order to get more insight in the traffic generated by the individual applications, we consider the requirements on the transfer of an single unit of information. [39] and [95] list for a number of applications the expected size of an information unit. Furthermore, they give an indication of the required response time for these applications. Figure 2-9 graphically represents these requirements. For each application an area of requirements is given, expressing both the uncertainty and the variation in requirements. Note that the information units sizes and response time requirements are given at the application level. An information unit may very well be transported in a number of smaller packets at the network level.

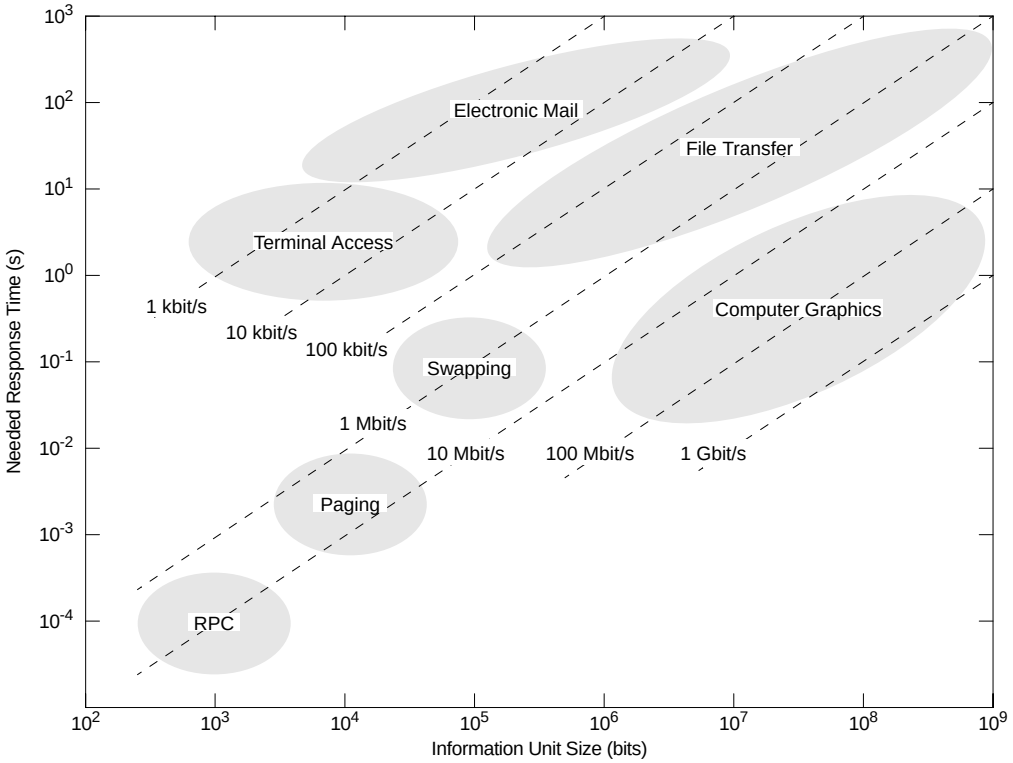


Figure 2-9: Applications Requirements (based on [39] and [95])

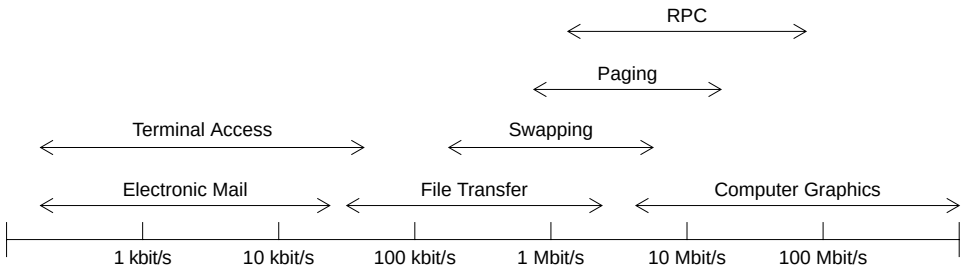


Figure 2-10: Throughput Requirements of Applications (based on [39] and [95])

In Figure 2-9 a number of diagonal lines have been drawn that give an indication of the throughput required to transfer information units of a certain size within the response time. Figure 2-10 summarizes these throughput requirements. It can be seen from the figure that throughput requirements range from less than a kilobit per second for electronic mail and terminal access to a gigabit per second for computer graphics.

In principle, the given applications do not tolerate any loss or corruption of data. This does not imply that no loss or corruption can be tolerated from the connectionless service we are designing. End-to-end protocols used on top of the connectionless service (e.g., a transport protocol) can enhance the provided QoS. Practical values for these and QoS requirements will be given in Section 3.1.

2.3 Prior and Related Work

Some of the related work, especially that published by standardization organizations or interest groups, has been a starting-point for our work. Other work has a different scope, or is complementary to our work.

The first work on integrating datagram (i.e., connectionless) and connection-oriented (e.g., voice) communications in a single network based on a fast packet switching technique, performed at AT&T Bell Laboratories, has been reported by Turner ([132], [133]). At the same time Coudreuse et al. of France Telecom worked on the asynchronous time division multiplexing technique ([23]). Eventually, these activities led to the definition of ATM, first within the Telecommunication standardization section of the International Telecommunication Union (ITU-T, the former CCITT) ([16], [68], [73]), and later on also in the ATM Forum ([4]).

ATM has been developed as a connection-oriented technique for, among others, the following reasons. Because of the required high-speed operation of the network, the time needed for processing in the nodes, e.g., for routing should be kept low. Furthermore, many applications require a guaranteed QoS, which can be individually negotiated for a specific application process. After the establishment of ATM, as a basically connection-oriented technique, the problem of supporting connectionless communications using ATM became of significant interest. A specification of the service to be provided to the user and the access to it has been given by a number of different organizations. Bellcore has defined the Switched Multi-megabit Data Service (SMDS), which should initially be provided using MANs and later using ATM ([6]). The European Telecommunications

Standards Institute (ETSI) has defined the european version of SMDS, called Connectionless Broadband Data Service (CBDS) in [35], [36], [37], and [38]. Finally, the ITU has defined the Broadband Connectionless Data Service (BCDS) to be provided using an ATM-based B-ISDN ([16], [19], [69]).

Currently, recommendations on how to provide the BCDS are being defined within the ITU ([20], [75], [76]). They mainly focus on the protocols to be used on top of ATM for the provision of the BCDS. Furthermore, architectural issues are considered. A general overview of these issues is given in [26], [111], [136], and [137].

A number of articles have been published related to the protocols defined in the ITU recommendations. [2], [86], and [130] deal with the design of AAL protocols to be used for data communications. Protocols to be used on top of the AAL are dealt with in [29], [126], and [135]. Finally, performance studies related to the design of these protocols have been presented in [15] and [34].

An environment where the use of a connectionless service on ATM can be expected to become prevailing is the Internet. The use of ATM for the transfer of packets for the Internet Protocol (IP) is being studied ([3], [21], [60], [92], [119]).

Many papers have been published on architectural aspects of providing a connectionless service using an ATM network ([11], [12], [29], [39], [64], [116], [128], [134], and [136]). Besides describing entities and protocols, most of these papers also identify different options for providing a connectionless service, such as the direct method vs. the indirect method, and message mode vs. streaming mode (see Chapter 3).

A significant contribution to the definition of protocols and interfaces between entities has been made by research projects in the RACE (Research and development in Advanced Communications technologies for Europe) program ([112], [113], [114]).

Relatively few papers report on the implementation of CLSs, i.e., the nodes installed in the network to route packets. [28] and [116] describe the design and implementation of a large scale connectionless server, connecting multiple ATM links, at the Alcatel Bell Telephone Research Centre. Design of small connectionless servers, attached to a single ATM link is described in [12], [61], [85], and [105].

Performance studies related to design and implementation options for connectionless servers are described in [8], [9], [13], [61], and [90]. These will be further elaborated on in Chapter 5.

The problem of the use of ATM connections and the reservation of bandwidth on these connections has received considerable interest. This has resulted in a large number of publications on the subject. A large number of these publications are authored by Gerla et al., and deal with a so-called bandwidth advertising algorithm ([25], [27], [40], [41], [42]). Others deal with mechanisms to modify bandwidth reservations during the lifetime of a connection on an on-off basis ([39], [103], [104], [118], [129], [141], [142], [145], [148]). Proposals to do this on a multi-level basis have also been done ([10], [32], [33], [91], [98], [120], [123], [147]). Most of these papers contain also some performance analyses of proposals.

2.4 Summary and Concluding Remarks

In this chapter, we have defined the notion of connectionless communications, as opposed to connection-oriented communications. Further, the reader has been introduced to the B-ISDN, a public telecommunication network aimed at providing a large base of applications to it users via an integrated access. Eventually, the communication needs of all these applications must be served by a single underlying infrastructure based on ATM. The basics of ATM are asynchronous time division multiplexing and fast packet switching.

ATM will also be used to provide a connectionless service to the users. Applications using this service have been identified, and basic requirements have been outlined. Most end-systems with these applications will access the B-ISDN via LANs or MANs. Throughput requirements of individual applications will range from less than a kilobit per second to values of the order of a gigabit per second.

A survey of related work that can be found in the literature has been given.

We have seen that ATM is a connection-oriented technique, where a connection is established prior to transferring fixed-size cells. Connection establishment requires knowledge about the traffic that is going to be transferred, in order to determine the bandwidth to be reserved. There is clearly a mismatch between ATM and the connectionless service some applications require. The connectionless service should transfer data units of any size. Moreover, it is essential to connectionless communications that no advance knowledge about the traffic to be transferred is exchanged with the system that provides the service. To bridge this gap between ATM and the connectionless service to be provided is the aim of this dissertation.

Does anyone need yet another blank skyscraper?
If you're like me I'm sure a minor miracle will do.

Lou Reed - New York

Chapter 3
Architectural Framework

The design of a complex communication system, such as the one this dissertation is concerned with, is a very complicated task. In order to be able to cope with this complexity, design methodologies have been developed, which allow for recursive decomposition of a system into subsystems that exhibit a lower complexity (e.g., in [139]). This chapter will use modelling concepts described in [122], [138], and [139], in order to come up with an architectural framework for the provision of connectionless communications over an ATM network. The architectural framework allows us to put the protocols that have (partially) been designed by different organizations in perspective. The various entities that are needed to provide the service are identified, their functionality is defined, and the way they interact with each other is specified.

We base our work on the above mentioned design methodology, because it uses a number of clear, well-defined concepts. Furthermore, the equivalence of the system in different stages of the design is more easily seen than with other methodologies, e.g., the one described in Recommendation I.130 of the ITU ([17]). As a result, the concepts of [139] provide a consistent and rigorous basis for the framework presented in this chapter.

Let us introduce the concepts used in the architectural framework. For a more detailed description of the concepts we refer to [122] and [139]. It should be noted that these concepts may be defined differently from what the user is familiar with, and that terms that are used to denote the concepts are often also used in a different context, with a slightly different meaning.

The first concept to be introduced is that of a *service*. A service specification is a description of the (required) external behaviour of a system. The (communication) system, or service provider, is regarded as a whole, no component parts nor their mutual interactions are visible. Users interact with the service provider at Service Access Points (SAPs). A service can be described by specifying these interactions, or service primitives, their parameters, and the possible sequences in which the primitives can be issued at the different SAPs. The service description

is at the highest level of abstraction. It does not prescribe how the interactions should take place.

A *protocol* specification is a description of a system as a collection of system parts. A protocol is a set of rules according to which subsystems (or protocol entities) cooperate to collectively provide a service. For this purpose, the protocol entities use the service provided by an underlying service provider.

By recursively decomposing a service provider in a protocol and an underlying service provider, a complex service can be decomposed into a number of layered protocols, until an underlying service is encountered that is already available.

A result of this recursive decomposition is a *network architecture*. It describes the protocol entities that can be identified in the system, the way they interact, and how they cooperate to provide certain services.

Let us finally introduce the notion of a *real system*. A real system is the implementation of a service provider or (a set of) protocol entities, i.e., compositions of software and hardware.

The architectural framework that is presented in this chapter defines the service that is required from the system under design, i.e., the system providing a broadband connectionless service to its users. It also describes a service that is already available for use in the design of this system, i.e., the service provided by an ATM network. Furthermore, the network architecture identifies intermediate services, the protocol entities that provide either the required or an intermediate service, and the protocols.

It should be noted that the protocols described in this chapter are not novel. They have been standardized, or proposed for standardization. However, the overall architectural framework in which the protocols are positioned is new.

The descriptions of the services and protocols are given in prose. It is beyond the scope of this dissertation to give formal specifications. A specification of the required service of the system for connectionless communications using ATM, in the formal specification language LOTOS ([67]) is given in [131].

In Section 3.1, the service description of the system to be designed will be given. In Section 3.2, the service provided by an ATM network will be described. The actual decomposition is performed in Section 3.3, which is the main body of this chapter. In this section, a network architecture for providing a broadband connectionless service using ATM is presented. Section 3.4 and Section 3.5 come up with

candidate protocols for the protocol layers identified in the network architecture, the ATM Adaptation Layer (AAL), and the Broadband Connectionless Network Layer. This chapter ends with a summary and concluding remarks, in Section 3.6.

3.1 The Broadband Connectionless Service (BCLS)

This section specifies the service that should be provided by a system for connectionless data communications over ATM. The service will be referred to as Broadband Connectionless Service (BCLS). The definition of the BCLS in the following subsections covers those aspects of the service that are relevant for this dissertation, given the level of detail of the design we are aiming at. It captures requirements that are expected to be posed on systems for broadband connectionless data communications. The service description functions as a starting point for the design in this dissertation.

Specifications of systems for broadband connectionless data communications have been given elsewhere. Bellcore specifies the Switched Multi-megabit Data Service (SMDS) in [6]. General features of SMDS and protocols for the interface to an SMDS system are defined. However, a service description as given here is not available for SMDS. The ETSI specifies the Connectionless Broadband Data Service (CBDS) in [35], [36], [37], and [38]. The CBDS is very similar to SMDS, and so is the specification. The ITU specifies a Broadband Connectionless Data Service (BCDS) in Recommendation F.812 ([19]), and some guidelines for the support of this service by B-ISDN in Recommendation I.364 ([76]). The service specification of Recommendation F.812 specifies the interface to a system providing a BCDS over B-ISDN in general terms. Recommendation I.364 specifies a protocol for this interface (the UNI). Both recommendations are not yet complete.

The service description presented here is based on [50], [112], and [131]. Most of the information has been taken from the SMDS specification, since the ITU recommendation on BCDS is far from complete. Conflicts of this service specification (and the SMDS specification) with the ITU recommendations will be explicitly mentioned.

We only describe the observable behaviour of the system, without describing how a UNI should be implemented. This is discussed later in this chapter, since we believe that it should be considered in a later design stage. Where necessary, the observable behaviour has been derived from the interface specification in [6] and [76].

In Section 3.1.1 we describe general aspects of the service. The service primitives that have been defined are presented in Section 3.1.2. The issue of QoS is discussed in Section 3.1.3. Section 3.1.4 gives some extra features of the service, called supplementary services, and Section 3.1.5 gives values for the QoS parameters.

3.1.1 General

The BCLS provides a means by which Service Data Units (BCL-SDUs) of variable but limited length are delimited and transparently transferred from one source Service Access Point (BCL-SAP) to one or more destination BCL-SAPs in a single service access, without establishing or later releasing a connection between source and destination BCL-SAPs.

Associated with the service are certain Quality of Service (QoS) parameters. It is assumed that the values of these parameters are fixed at the time of subscription to the service, or negotiated by management procedures. Alternatively, values for some of the parameters could be determined per transferred BCL-SDU. This option has been defined in the ITU BCDS. SMDS on the other hand does not allow this. We assume in this dissertation that QoS cannot be selected on a per-SDU basis, but only per service subscription.

The BCLS can also provide some extra features, called supplementary services (see Section 3.1.4). The provision of these features is negotiated at the time of subscription to the service.

3.1.2 Service Primitives

Two service primitives are needed to provide the BCLS:

- BCL-DataRequest, with parameters
  Source Address, Destination Address, and BCL-SDU; and
- BCL-DataIndication, with parameters
  Source Address, Destination Address, and BCL-SDU.

The BCL-SDU parameter, which contains the user data, may have any length up to a certain maximum. This maximum length has been defined to be 9188 octets.

The Source Address parameter refers to the BCL-SAP at which the BCL-DataRequest primitive is issued. The Destination Address parameter refers to either an individual BCL-SAP or a number of BCL-SAPs in case of multicast.

In general, a BCL-DataRequest primitive at a certain BCL-SAP, corresponding to the Source Address parameter, will result in a BCL-DataIndication primitive at the BCL-SAP corresponding to the Destination Address parameter. If the Destination Address parameter refers to a number of BCL-SAPs, the BCL-DataRequest primitive will result in a BCL-DataIndication primitives at all BCL-SAPs (multicast). The value of all three parameters will be the same for a BCL-DataRequest primitive and the corresponding BCL-DataIndication primitive(s). For a given source-destination pair of BCL-SAPs, a BCL-DataRequest primitive will result in a BCL-DataIndication primitive before any subsequent BCL-DataRequest primitive results in a BCL-DataIndication primitive, i.e., there is in-sequence delivery.

With a certain (low) probability, the BCL service provider may exhibit a slightly different behaviour. The value of this probability depends on the agreed QoS. The following cases of exceptional behaviour are identified:

- loss
  The BCL-DataRequest primitive does not result in a BCL-DataIndication primitive.
- duplication
  The BCL-DataRequest primitive results in two BCL-DataIndication primitives at the same BCL-SAP.
- misdelivery
  The BCL-DataRequest primitive results in a BCL-DataIndication primitive at a BCL-SAP not specified in the destination address parameter.
- corruption
  The BCL-SDU parameter of a BCL-DataIndication primitive differs from the BCL-SDU parameter of the corresponding BCL-DataRequest.
- missequencing
  For a given source-destination pair of BCL-SAPs, a BCL-DataRequest primitive results in a BCL-DataIndication primitive which happens later than the BCL-DataIndication primitives corresponding to one or more subsequent BCL-DataRequest primitives.

3.1.3 Quality of Service

The term Quality of Service (QoS) refers to certain characteristics of a connectionless transfer as observed between the BCL-SAPs. QoS parameters will be fixed at

the time of subscription or negotiated by management procedures during service provision.

The following QoS parameters can be identified:

- maximum throughput
  The maximum number of bits (contained in BCL-SDUs) that may be offered by a service user to the service provider per time unit.
- transit delay
  The elapsed time between BCL-DataRequest primitives and the corresponding BCL-DataIndication primitives.
- loss probability
  The probability that a loss occurs, estimated by the ratio of lost BCL-SDUs to the total number of transferred BCL-SDUs.
- duplication probability
  The probability that a duplication occurs, estimated by the ratio of duplicated BCL-SDUs to the total number of transferred BCL-SDUs.
- misdelivery probability
  The probability of misdelivery, estimated by the ratio of misdelivered BCL-SDUs to the total number of transferred BCL-SDUs.
- corruption probability
  The probability that a corruption occurs, estimated by the ratio of corrupted BCL-SDUs to the total number of transferred BCL-SDUs.
- missequencing probability
  The probability of missequencing, estimated by the ratio of missequenced BCL-SDUs to the total number of transferred BCL-SDUs.

3.1.4 Supplementary Services

The term Supplementary Services refers to some extra features that can be provided by the BCLS to extend its functionality. The provision of these supplementary services will either be fixed at the time of subscription, or negotiated by management procedures during service provision.

The following supplementary services have been identified:

- Closed User Group (CUG)
  This supplementary service enables users to form groups, to and from which access is restricted. A specific user may be a member of one or more closed user group. Closed user groups may have additional capabilities whereby individual members may send data to users outside the group, and/or receive data from users outside the group, e.g., in Virtual Private Networks (VPNs).

- Address Screening
  This supplementary service ensures that a user cannot or can only receive data from certain users and that a user cannot or can only send data to certain users. Address screening may be applied to individual addresses as well as to group addresses. Source address screening imposes restrictions on the set of users that are allowed to send to a specific destination. Destination address screening restricts the set of users to which a certain source may send. Address screening may be used for security reasons. It can also be used to create closed user groups.

3.1.5 QoS Values

As stated in Section 3.1.3, values of QoS parameters are not a part of this service specification. However, in order to have target values that can be used in performance studies, we list some values as they are specified for SMDS in [6] in Table 3-1.

QoS parameter	value
maximum throughput	4 - 34 Mbits/s
transit delay (single destination)	95% within 20 - 140 ms (depending on the type of UNI)
transit delay (multiple destinations)	95% within 100- 220 ms (depending on the type of UNI)
loss probability	$1x10^{-4}$
duplication probability	$5x10^{-8}$
misdelivery probability	$5x10^{-8}$
corruption probability	$5x10^{-13}$
missequencing probability	$1x10^{-9}$

Table 3-1: Values for QoS Parameters in SMDS

3.2 The ATM Service

Let us now describe the service provided by an ATM network. In the B-ISDN Protocol Reference Model (see Section 2.1.2) and in the protocols defined for B-ISDN, a distinction is made between a user plane and a control plane[1]. The user plane contains the functionality for transferring information on behalf of the user, given the presence of appropriate connections. The control plane contains functionality for the establishment, modification, and release of these connections. The functionality of the control plane is comprised in the signalling system.

For the transfer of BCL-SDUs, we are interested in the composite behaviour of the ATM layer in the user plane and the part of the signalling system controlling this ATM layer ([73]). A user of an ATM network willing to transfer data, should first set up an ATM connection via the control plane, transfer the data via the user plane, and finally release the connection again via the control plane. Within the ITU recommendations, no description of the composite behaviour of user plane and control plane is provided. A first proposal for such a description has been given in [55]. However, this proposal describes the behaviour of a B-ISDN system at a much higher layer, i.e., at the layer where multi-media services are supported.

Two major reasons prevent us from coming up with an ATM service description specifying the composite behaviour of user and control plane. The first one is that in B-ISDN no distinction is made between control of the ATM layer and control of higher layers, so that it is somewhat artificial to attribute part of the control plane to the ATM service. The second one is that the specification of control protocols for B-ISDN is still an ongoing process, where the functionality of the protocols is gradually extended by defining protocols for different capability sets ([79], [80]). Our objective with respect to ATM is to describe the external behaviour of a given system, not to specify the requirements on a system under design. Therefore, the description provided here gives general characteristics of the ATM service as far as control aspects are concerned, and a more detailed specification for the transfer of user information over ATM connections.

In Section 3.2.1 we describe some general aspects of the ATM service. Characteristics of the service, related to the control of connection, are given in Section 3.2.2. The service primitives that have been defined for the transfer of user information are given in Section 3.2.3. Related to a certain ATM connection are traffic parame-

[1] Note that a management plane has been identified as well. The management plane is not considered in this context.

ters and QoS parameters, describing characteristics of the traffic generated by the user, and the way the ATM service transfers the traffic. These are discussed in Section 3.2.4 and Section 3.2.5. Finally, in Section 3.2.6, QoS parameters related to the control of connections are discussed.

3.2.1 General

The ATM service provides a means by which ATM Service Data Units (ATM-SDUs) of a fixed, standardized length are delimited and transparently transferred from a single source ATM-SAP to one or more destination ATM-SAPs along a previously established connection. Establishment and release of a connection can be done either on demand or (semi-)permanently by management procedures.

Associated with the service are certain QoS parameters. The values of these parameters are negotiated during the connection establishment. Furthermore, traffic parameters are associated with an ATM connection, specifying characteristics of the traffic a user offers to the network for transfer over the connection. These parameters are negotiated during connection establishment and can be renegotiated during the lifetime of the connection.

3.2.2 Control of ATM Connections

ATM-SDUs can only be transferred from a source ATM-SAP to one or more destination ATM-SAPs if an ATM connection from the source SAP to the destination SAP(s) is available. A service user has access to the connection at an ATM Connection Endpoint (ATM-CEP). The presence of the connection corresponds to the presence of a source CEP and one or more associated destination CEPs. An ATM connection has a source CEP in the source SAP, and a destination CEP in each destination SAP.

Two types of connections have been identified, Virtual Channel Connections (VCCs) and Virtual Path Connections (VPCs). An ATM-SDU transferred over a VPC has associated with it an extra identifier (the VCI), which can be used by the user of the ATM service to identify the SDU for its own purposes. The VCI is transferred transparently over a VPC.

Associated with connections of both types are QoS parameters (Section 3.2.5), and a set of traffic parameters (Section 3.2.4). Both give characteristics of the communication over the connection, and are the result of negotiation between the service users and the ATM network.

ATM connections can be available on demand, or on a (semi-)permanent basis. The installation of (semi-)permanent connections (either VCC or VPC) is taken care of by management. The traffic parameters of a (semi-)permanent connection can be negotiated and changed during the lifetime of the connection.

The signalling system of the ATM network is responsible for establishing on-demand connections. During the establishment, which is initiated by one of the service users, QoS parameters and traffic parameters are negotiated. The traffic parameters can be renegotiated during the lifetime of a connection, the QoS cannot.

QoS parameters are also associated with the signalling itself. They give characteristics of the control of connections. These are elaborated on in Section 3.2.6.

3.2.3 Service Primitives

Service primitives are always associated with a single connection. They are issued at a certain CEP in a SAP. The following two service primitives have been defined for the transfer of user information over the ATM service:

- ATM-DataRequest, with parameters
  ATM-SDU, Submitted Loss Priority, Congestion Indication, ATM-user-to-ATM-user Indication, and VCI (only for a VPC); and

- ATM-DataIndication, with parameters
  ATM-SDU, Congestion Indication, ATM-user-to-ATM-user Indication, and VCI (only for a VPC).

The ATM-SDU parameter, which contains the user data, has a length of exactly 48 octets. It is passed transparently by the ATM network from source to destination(s) along the ATM connection. The Submitted Loss Priority parameter indicates the importance of the ATM-SDU relative to other SDUs transferred over the same connection. It is either high or low. The Congestion Indication parameter is used to indicate that the ATM-SDU has passed through a congested network node. Finally, the ATM-user-to-ATM-user Indication (AUU) parameter is transparently passed by the ATM network, just like the ATM-SDU. Its length is a single bit. The VCI is only a parameter if the CEP where the primitive is issued is the endpoint of a VPC. It is passed transparently by the ATM network, and can be used by the service user as an extra identification for the ATM-SDU.

Given the availability of a connection, the general operation of the ATM service is as follows. An ATM-DataRequest primitive at a certain source CEP in a SAP will

result in an ATM-DataIndication primitive at the destination CEP associated with the source CEP. If several CEPs are associated with the source CEP, an ATM-DataIndication primitive will result at all destination CEPs. The ATM-SDU and the AUU parameters are identical for the ATM-DataRequest and the corresponding ATM-DataIndication(s). If the Congestion Indication Parameter was not set in the request primitive, it may be either set or not set in the indication primitive. If it was set in the request primitive, it will be set in the indication primitive as well. For a given pair of CEPs, the sequence of ATM-DataIndication primitives is identical to the sequence of the corresponding ATM-DataRequest primitives.

With a certain (low) probability, the ATM service provider may exhibit a slightly different behaviour. The value of this probability depends on the QoS agreed for the connection. The following cases of exceptional behaviour are identified:

- loss
  The ATM-DataRequest primitive does not result in an ATM-DataIndication primitive.
- misinsertion
  An ATM-DataIndication primitive occurs at a CEP, while no ATM-DataRequest has been issued at an associated source CEP.
- corruption
  The ATM-SDU parameter of an ATM-DataIndication primitive differs from the ATM-SDU parameter of the corresponding ATM-DataRequest.

3.2.4 Traffic Parameters

Traffic parameters are used to describe traffic characteristics of an ATM connection. They characterize the traffic that is offered by the service user to the network. These parameters are negotiated during the establishment of a connection, and may be renegotiated during the lifetime of a connection.

Currently the ITU specifies only a single traffic parameter:

- peak cell rate
  The peak cell rate is defined as the inverse of the minimum time elapsing between two consecutive ATM-DataRequest primitives.

Other parameters, such as the mean cell rate, are being studied.

3.2.5 Quality of Service of an ATM Connection

A number of QoS parameters are associated with an ATM connection. They can be negotiated during the establishment of the connection. The agreed QoS parameters will only be met by the network if the user confines to the agreed traffic parameters. Up to now only two QoS parameters have been identified within the ITU. They, however, have not yet been defined exactly:

- cell delay variation
  The cell delay variation is the deviation of the actual cell delay from the mean cell delay experienced for the connection.

- cell loss ratio
  The cell loss ratio is the ratio of lost ATM-SDUs to the total number of transferred ATM-SDUs on a connection. For a single ATM connection, two cell loss ratio objectives can be defined, one for ATM-SDUs with a high submitted loss priority parameter and one for those with a low submitted loss priority parameter.

3.2.6 Quality of Service of the Signalling System

Other relevant QoS parameters for the ATM service are those characterizing aspects of the control of connections. These have not been defined by the ITU yet. In the RACE MAGIC project, QoS parameters related to the entire control plane of the B-ISDN are defined ([115]). In [62], a method for developing performance measures related to these parameters, has been proposed.

We are only interested in QoS parameters quantifying behavioural aspects of the control plane as far as the control of ATM is concerned. For the performance studies in this dissertation, we need to make assumptions about the following parameters:

- connection establishment delay
  The connection establishment delay is the time elapsing between the request of a user to the signalling system to establish a connection, and the confirmation from the signalling system that the connection is available.

- traffic contract renegotiation delay
  The traffic contract renegotiation delay is the time elapsing between a request from a user of a connection to modify the traffic contract, and the confirmation from the network that the new contract is in force.

3.3 Network Architecture

Now that we have defined the Broadband Connectionless Service in Section 3.1, we have the starting point for the decomposition of the system under design in subsystems. The network architecture that results from this decomposition specifies the types of protocols that are needed to provide the BCLS, the protocol entities that implement the protocols, and the underlying services that are used by these entities. We present the network architecture in a stepwise approach, where the steps correspond to subsequent decompositions of the service. In the past, similar decomposition techniques have been successfully applied to other problems ([109], [121]).

In this section, we will first introduce the notation that is used to present the network architecture, in Section 3.3.1. Basically, there are two methods of providing a connectionless service over ATM. We introduce them in Section 3.3.2. The subsequent decompositions of the BCLS are presented in Section 3.3.3 for the first method, and in Section 3.3.4 for the second method. An overall view of the network architecture for both methods will be given in Section 3.3.5. In Section 3.3.6, we will derive a protocol reference model from these network architectures. Finally, in Section 3.3.7, we relate the architectural framework given in this chapter to architectures and models defined in standards.

3.3.1 Notation

Let us first explain the notation that is used to describe the network architecture in a symbolic way (see Table 3-2). We present the network architecture by showing the decomposition steps that are taken. In each figure (Figure 3-1 through 3-12) both the unpartitioned and the partitioned (sub)system are shown. The unpartitioned (sub)system is shown on the left, the (sub)system after the decomposition step is shown on the right. The gray arrow (a) denotes this decomposition step. A system is shown in two different representations, one in the upper part of the figure, and one in the bottom part. The bottom part represents the actual network architecture by displaying protocols, subsystems, protocol entities, and underlying services. The upper part is displayed for illustration. It gives a representation of a real system that can be associated with the network architecture. Dashed lines (b) show the correspondence between the two representations.

For the description of the network architecture we use the following notation. A (sub)system of which only the external behaviour is considered (a service

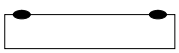
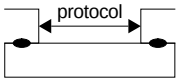
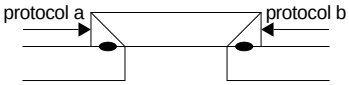
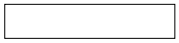
	(a): decomposition step
	(b): corresponding representations
	(c): service provider
	(d): cooperating protocol entities
	(e): protocol conversion
	(f): end-system
	(g): interconnection
	(h): intermediate system
	(i): network

Table 3-2: Notation for the Symbolic Description of a Network Architecture

provider), is represented by a rectangle (c). The black ovals denote the SAPs at which service users interact with the service provider. The cooperation between a number of subsystems (e.g., protocol entities) according to the rules of a certain protocol, using an underlying service provider, is represented by a double arrow (d). Adjacent rectangles are consider as subsystems that can directly interact. Subsystems are assumed to cooperate with other subsystems using a single protocol. If they are not, i.e., if some protocol conversion has to be performed, the subsystem is drawn with diagonal lines in it (e).

For the representation of a real system, four symbols will be used. They represent real systems that are end-systems from the point of view of the BCLS (f), interconnections at a lower layer (g), intermediate systems (h), and networks (i).

3.3.2 Methods of Providing a Connectionless Service

Within the ITU, two methods have been identified to provide a connectionless service using an ATM-based B-ISDN ([69]). These have been characterized as follows[2]:

- “Indirectly via a B-ISDN connection-oriented service:
  In this case a transparent connection of the ATM layer, either permanent, reserved or on demand, is used between B-ISDN interfaces.”

- “Directly via a B-ISDN connectionless service:
  In this case the connectionless service function would be provided within the B-ISDN.”

The indirect method uses end-to-end ATM connections to connect a pair of users that want to communicate connectionless. The direct method use special nodes in the B-ISDN, called Connectionless Servers (CLSs), to which users that want to communicate connectionless are connected by means of ATM connections. These two methods of providing a connectionless service over ATM are not mutually exclusive. Both methods can be used simultaneously over the same ATM network. Furthermore, a single user can use both methods, e.g., depending on the destination.

The two ways of providing a connectionless service are reflected in the network architecture. Let us first present the decomposition of the BCLS that corresponds to the indirect method.

3.3.3 Indirect Method

The BCLS can be provided on top of an end-to-end connection-oriented (CO) network (Figure 3-1). A protocol is needed that relates the required destination address for a BCL-SDU to a SAP of the underlying network, and routes a PDU containing the SDU on a connection to that SAP. A connection management function is needed in the protocol to request the underlying network to establish, maintain and release end-to-end connections between the protocol entities in the end-systems when needed. This protocol will be referred to as Broadband Connectionless Network Protocol (BCL NP).

Note that the term end-system refers to a real system that is an end-system with respect to the B-ISDN, i.e., to a node that is connected to the B-ISDN via its User Network Interface (UNI). According to the B-ISDN Reference Configuration, described in [78], this is a Terminal Equipment (TE) or Network Terminator type 2 (NT2). However, it can be an intermediate system in an internet, e.g., an interworking unit to a LAN or MAN.

---

[2] Depending on the intuition of the reader, the terms indirect and direct, which have been introduced by the ITU, can be misleading and easily confused.

Figure 3-1: Decomposition of the BCLS using the Indirect Method

The end-to-end CO network should provide a means by which SDUs of a variable size can be transferred between end-systems over a previously established connection. In an ATM environment, this service can be provided by an ATM Adaptation Layer (AAL) protocol, using the ATM service as the underlying service (Figure 3-2). Such an AAL protocol is also an end-to-end protocol. It segments SDUs of a variable length into fixed size ATM-SDUs, and reassembles the variable size SDUs again at the destination.

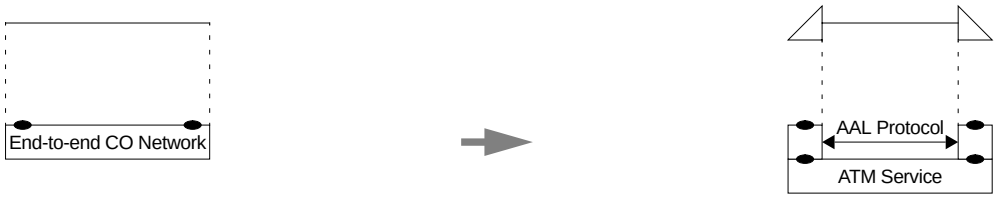


Figure 3-2: Decomposition of the End-to-end CO Network

The ATM service has been described in Section 3.2. Although the ATM service is taken as a starting point for the design, we include its decomposition in the network architecture. This is to obtain an overall framework of protocols and entities needed for the provision of the BCLS and to be able to identify which protocols need to be implemented in which real system.

The ATM service is provided by an ATM network to which end-systems have access over a UNI (Figure 3-3). This UNI is indeed the one that has been specified for the B-ISDN. The ATM protocol to be used over this UNI has been specified in [73]. The underlying Physical Layer (UNI PL) service has been defined in [18].

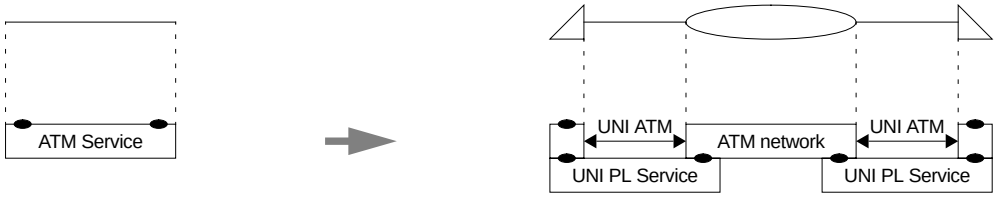


Figure 3-3: Decomposition of the ATM Service

The ATM network itself can be seen as composed of ATM nodes, which are in fact the exchanges of the B-ISDN. Within the ITU, the interface between these exchanges is called the Network Node Interface (NNI). The ATM nodes cooperate according to the ATM protocol that has been defined for the NNI ([73]). For their communication, they use the service provided by the physical layer, as it has been defined for the NNI (NNI PL) (Figure 3-4). Note that only in some of the ATM nodes (local exchanges) interworking with UNI protocols has to take place. The other nodes (transit exchanges) just have to implement NNI ATM protocols.

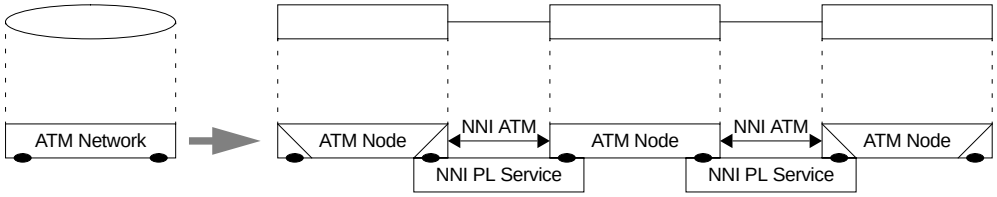


Figure 3-4: Decomposition of the ATM Network

3.3.4 Direct Method

In the network architecture presented in the previous section, the BCL network protocol was only implemented in the end-systems. In the network architecture that will be presented now, the network will also implement this protocol. This network architecture corresponds to the so-called direct method of providing a connectionless service.

If the direct method is used, the BCLS will be provided to the user by some functionality in an end-system, which is cooperating with the BCL Network (Figure 3-5). The end-system and the BCL Network cooperate according to a BCL Network Access Protocol (BCL-NAP) using a BCL Access Network as an underlying service. The protocol entity in the end-system will pack a BCL-SDU, received from the user, in a PDU together with the source and destination address, and pass it to the BCL Network. Subsequently, the BCL Network will transport the PDU, and pass it to the appropriate remote user entity. In order to enable the passing of PDUs between a user entity and the BCL Network, the BCL Access Network provides for the communication between the cooperating entities. This access network can be connection-oriented or connectionless. If it is connection-oriented, a connection between the user entity and the BCL Network needs to be established before communication is possible.

Not all users will access the BCL Network using the same BCL-NAP, and using the same protocols for the BCL Access Network. The access network can for

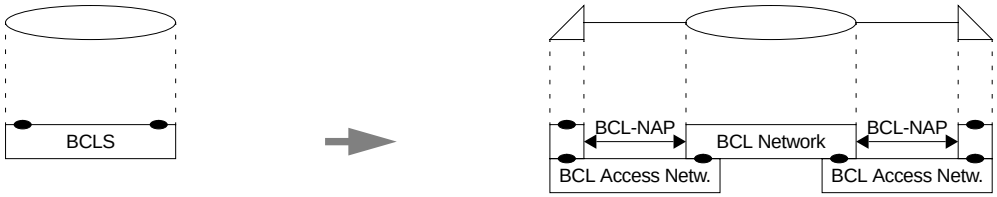


Figure 3-5: Decomposition of the BCLS using the Direct Method

instance be based on ATM, or on the DQDB protocols ([63]). In this heterogeneous environment, the BCL-NAPs will not be exactly the same, but they will provide roughly the same functionality, e.g., the same addressing conventions have to be used by both protocols. For that reason, PDUs from one protocol can be mapped on PDUs from the other, quite easily. Figure 3-6 represents the decomposition of the BCLS in case of heterogeneous access networks.

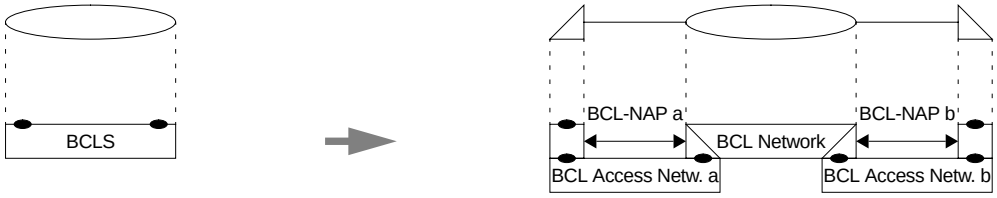


Figure 3-6: Decomposition of the BCLS with Heterogeneous Access Networks

Let us first consider the access network to the BCL network. Two candidates for the access protocol are ATM and DQDB. The choice for a particular underlying protocol has an impact on the structuring of the access network. An underlying protocol system can be thought of as a predefined building block, with a predefined functionality. The other building blocks have to match this functionality. Since this dissertation is concerned with the transfer of connectionless data, using ATM, we will focus on BCL Access Networks that have ATM as an underlying protocol. Such an access network has a connection-oriented nature.

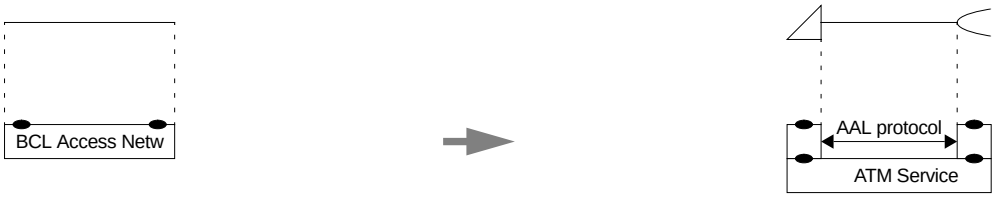


Figure 3-7: Decomposition of the BCL Access Network

The BCL Access Network should be able to transfer BCL-NAP PDUs of a variable size. In order to adapt to the fixed size of ATM-SDUs, the BCL Access Network is decomposed into cooperating protocol entities, taking care of segmentation and

reassembly of BCL-NAP PDUs, and an underlying ATM service (Figure 3-7). The protocol used for segmentation and reassembly is an AAL protocol.

The ATM service provider is very similar to the ATM service defined for the indirect method of providing connectionless services. The difference here is that the access of the BCL Network to the ATM network is through an NNI instead of through a UNI. This yields the following decomposition (Figure 3-8).

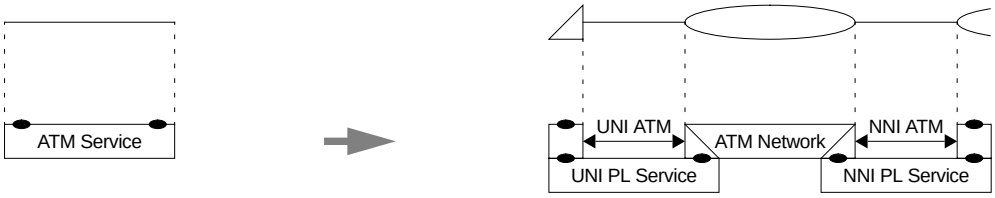


Figure 3-8: Decomposition of the ATM Service for the BCL Access Network

Now that we have presented the decomposition of the access network to the BCL Network, let us present the decomposition of the BCL Network itself. A BCL Network consists of a number of protocol entities, which cooperate according to a Broadband Connectionless Network Protocol (BCL NP), and which communicate with each other using the service of an Interconnection Network (Figure 3-9).

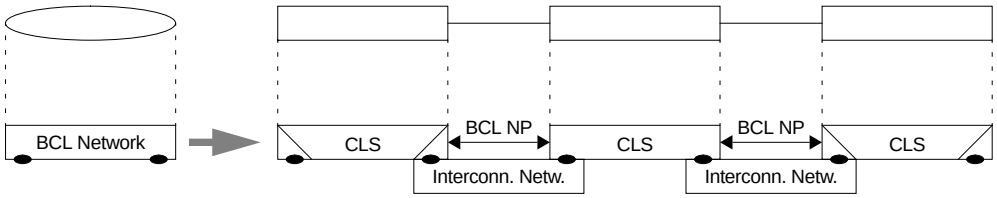


Figure 3-9: Decomposition of a BCL Network

These protocol entities route PDUs containing a BCL-SDU including source and destination address from source to destination. The PDUs are transferred between two entities along a connection established between SAPs in the Interconnection Network. The establishment and maintenance of connections, in order to connect protocol entities, is also a function of the BCL NP. A real system that implements the protocol entity of the BCL NAP and/or BCL NP together with the protocol entities of underlying protocols, is called a Connectionless Server (CLS). Two types of CLSs can be identified depending on whether or not it directly communicates with an end-system. An Access CLS does communicate directly with end-systems, and should thus implement both the BCL NP and the BCL NAP and perform conversion between the two. A Transit CLS does not, and

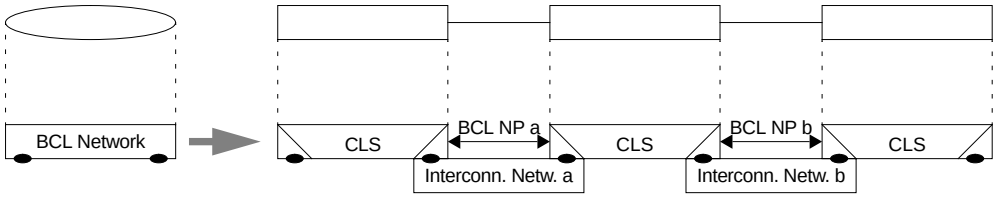


Figure 3-10: Decomposition of a BCL Network with Heterogeneous Interconnection Networks

should only implement the BCL NP. Transit CLSs will normally only be present in very large networks.

The Interconnection Networks that are identified in the last decomposition step are not necessarily of the same type. One could for instance be based on ATM while another is based on DQDB. This would imply that the service provided by the Interconnection Network is not the same, and thus that the BCL NPs used over this service are slightly different to adapt to those changes. Figure 3-10 shows the decomposition of the BCL Network in case of heterogeneous Interconnection Networks.

The Interconnection Network can be very diverse in nature. It can be a simple point-to-point link, with a datalink protocol running on top of it. It can also be a multi-access network like DQDB. We focus on the case where the Interconnection Network is based on ATM. We can further decompose the Interconnection Network into entities taking care of segmentation and reassembly, using the ATM Adaptation Layer protocol, and an underlying ATM service for the transfer of cells (Figure 3-11).

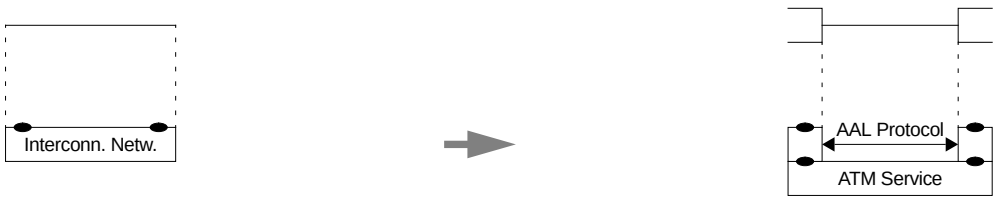


Figure 3-11: Decomposition of an ATM-based Interconnection Network

The ATM service is very similar to the ATM service defined for the BCL Access Network. Here all users of the ATM service access the ATM Network via an NNI. This yields the following decomposition (Figure 3-12).

The ATM network that is obtained from the decomposition of the BCL Access Network and the Interconnection Network, can be decomposed in the same way as the one that is obtained during the decomposition for the indirect method of

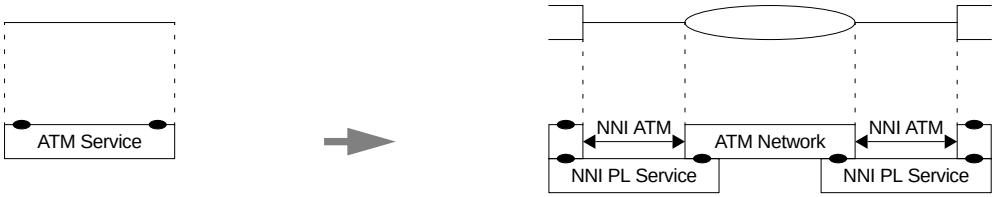


Figure 3-12: Decomposition of the ATM Service for the Interconnection Network

connectionless service provision. These ATM networks can in fact be realized by the same real network. Thus, ATM connections can be used by end-systems to access the BCL Network and by CLSs to connect to each other. In fact, the same real network can also realize the ATM network identified in Section 3.3.3, thus allowing for the coexistence of the direct and indirect method of providing connectionless services over the same ATM network.

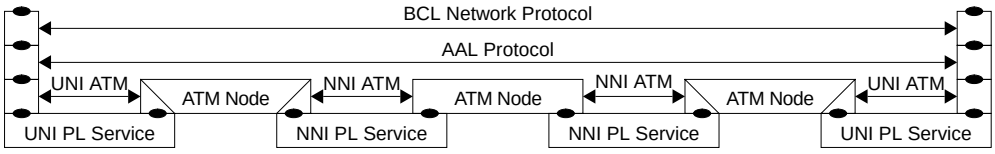


Figure 3-13: Network Architecture for the Indirect Method

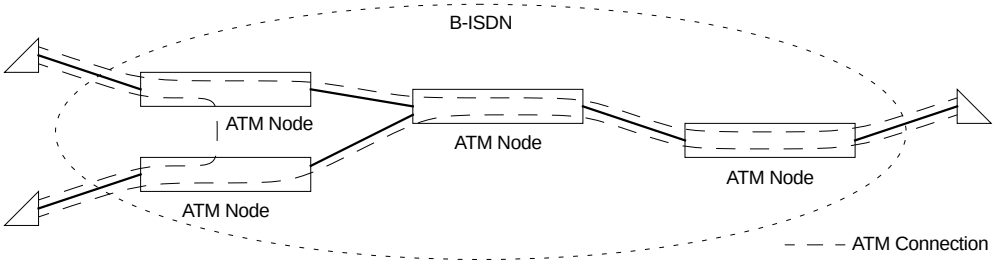


Figure 3-14: Example Network Configuration for the Indirect Method

3.3.5 Overall View on the Network Architecture

Let us try to summarize the results from the previous sections, in order to obtain an overall view of the network architecture. We present two figures that put all the mentioned protocols in perspective, one for the indirect method, and one for the direct method. Furthermore, we illustrate the network architectures by giving possible configurations of real systems that provide the connectionless service.

In case of the indirect method, the functionality of the BCL NP and the AAL protocol is only present in the end-systems (Figure 3-13). The intermediate systems contain only ATM and Physical Layer functionality. As a consequence,

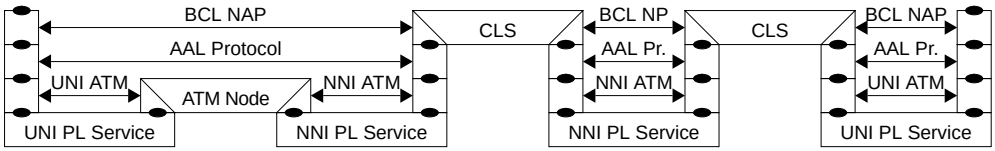


Figure 3-15: Network Architecture for the Direct Method

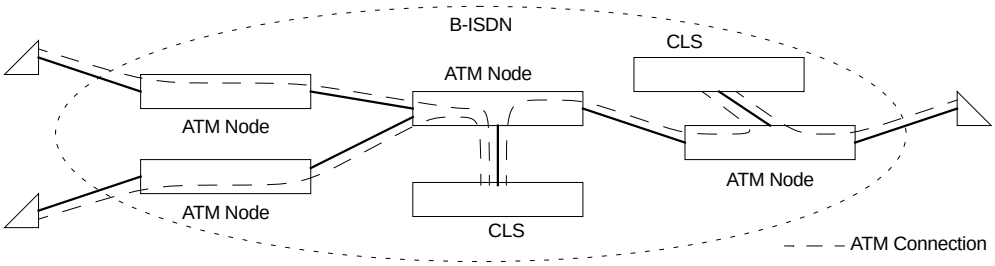


Figure 3-16: Example Network Configuration for the Direct Method

only ATM nodes are present in the network. Figure 3-14 shows a possible network configuration of an ATM-based B-ISDN, where the indirect method of providing a connectionless service is employed. All end-systems that wish to communicate need to be connected by ATM connections. These connections are switched by the ATM nodes in the B-ISDN.

In case of the direct method, BCL NP and AAL protocol functionality is present in intermediate systems within the B-ISDN, i.e., in CLSs (Figure 3-15). The CLSs communicate via ATM nodes, according to the BCL NP. Entities in the end-systems cooperate with a CLS according to a BCL NAP, probably also via ATM nodes. Figure 3-16 shows a possible configuration for providing a connectionless service using the B-ISDN according to the direct method. The two CLSs in this figure are connected by means of an ATM connection. End-systems that wish to communicate use an ATM connection to a CLS to transfer their PDUs.

3.3.6 Protocol Reference Model

In the network architectures, presented in the previous sections, a number of protocols have been identified for different parts of a system providing a BCLS. Some of these protocols have much in common, their major difference being whether they are used for access to a network or within a network. Furthermore, the protocols use the service provided by other protocols to communicate. Here, we define a layered model, a protocol reference model. Protocols with a similar functionality reside in the same layer. Protocols that are used by protocols from a specific layer for their communications reside in the next lower layer.

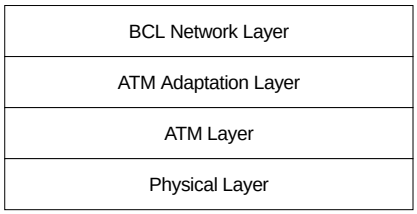


Figure 3-17: BCLS Protocol Reference Model

Figure 3-17 shows the BCLS Protocol Reference Model. The following four layers can be identified: BCL Network Layer, ATM Adaptation Layer, ATM Layer, and Physical Layer. Below we will discuss these layers in detail.

BCL Network Layer

The BCL Network Layer contains those protocols that are concerned with the end-to-end transfer of individual BCL-SDUs between source and destination. Furthermore, this layer should adapt the connectionless nature of the service to be provided to the connection-oriented nature of the services provided by protocols residing in the lower layers. The key functions to be performed in this layer are the following:

- Routing
  Individual PDUs are routed through the network, based on the address information contained in the PDU, and network information available in the network nodes.

- Connection Management
  In order to adapt the connectionless service to be provided to the connection-oriented nature of lower layer protocols, a function is needed that initiates connection establishment, modification, and release. This connection management function is responsible for connections between network protocol entities. Thus, the connection management function manages a network of connections between network nodes, called a *connectionless overlay network*. Managing this connectionless overlay network includes establishing connections and negotiating a suitable QoS and suitable traffic parameters. Furthermore, it includes modifying the traffic parameters of a connection when needed, or releasing the connection when it is no longer needed. For this purpose the connection management function requests the signalling system to perform the proper actions.

- Address Validation
  In the BCL Network, the source address specified by the sending user in the PDU is checked against the source addresses that are allowed for the AAL connection on which the PDU arrives.

- Access Class Enforcement
  The BCL Network enforces an end-system to comply with certain parameters (the access class) regarding the stream of packets that is sent into the network. The parameters are agreed upon at the time of subscription.

- Address Screening
  In order to allow for the creation of closed user groups, the destination address of PDUs is checked against a list of allowed addresses for the specified source (destination address screening). Furthermore, the source address of PDUs is checked against a list of allowed addresses for the specified destination (source address screening).

Protocols that reside in the BCL Network Layer are the BCL Network Access Protocol and the BCL Network Protocol.

ATM Adaptation Layer

Protocols for the ATM Adaptation Layer (AAL) provide for the transfer of AAL-SDUs of a variable but limited length along a connection from a source network layer entity to one or more destination network layer entities. The major function of these protocols is to adapt the variable length of AAL-SDUs to the fixed length of the ATM-SDUs.

The following key functions are performed:

- Segmentation and Reassembly
  An AAL SDU can have any length (up to a certain maximum). It is transferred using a number of AAL PDUs with a size of 48 octets. Protocol Control Information (PCI) is added during the segmentation, in order to enable correct reassembly at the receiver side.

- Error detection
  Bit errors in the AAL PDU can cause malfunctioning of the reassembly function, or the routing function in the BCL Network Layer. As a result of this, data could unintentionally be delivered to the wrong end users terminal. From a security point of view, this is a very undesirable situation. Therefore, detection of errors will be performed in the AAL.

ATM Layer

The ATM Layer contains protocols that provide for the transfer of fixed size ATM-SDUs between AAL entities along a previously established connection. In order to do this, the ATM-SDUs are transferred in ATM-PDUs (cells) from node to node using the Physical Layer service.

The following key functions are performed by the ATM layer:

- Identification of cells
  Cells, transferred using the Physical Layer service need to be identified, in order to know the virtual connection they belong to.
- Relaying of cells
  In all intermediate ATM nodes, incoming cells are relayed to a certain outgoing link, based on the virtual connection they belong to.

Physical Layer

The Physical Layer provides for the transfer of cells between ATM nodes. The key function of this layer is:

- Transmission of cells
  The ATM cells are transformed to a physical representation by the transmitting node. After propagation through the medium, the receiving node transforms the physical signals again to the original cell.

3.3.7 Relation to other Models and Architectures

A network architecture like the one given here, is not available in the literature. Some models and architectures can be found however. The ITU has defined a reference configuration for providing connectionless services ([76]). It locates so-called ATM switched capabilities and connectionless service functions relative to the B-ISDN. These can be seen as functions of the physical and ATM layer and functions of the AAL and BCL network layer respectively. This reference configuration further identifies the interfaces between the real systems that are needed to provide the service. Also in [76], the ITU gives a protocol architecture that puts the protocols to be used at the UNI in relation with each other.

The BCLS protocol reference model is different from the B-ISDN protocol reference model defined in [18]. The latter considers the transfer of user information and the control of connections as conceptually different functions, i.e., it distin-

guished between a user and a control plane, while it is useful that they are initially looked at from an integrated perspective, describing the required overall behaviour of the system under design. The B-ISDN protocol reference model only considers layers up to the AAL. The other layers are referred to as higher layers. The control functions residing in the ATM, AAL, and network layer of our model would reside in the higher layers of the control plane. The other functions of the network layer reside in the higher layer of the user plane.

The SMDS specification ([6]) does not provide protocol reference models or an architecture as presented here.

3.4 AAL Protocols

In Section 3.3.6, segmentation, reassembly, and error detection have been identified as the key functions to be performed in the ATM Adaptation Layer. An AAL protocol should provide these functions. It should adapt the service provided by the ATM service over an ATM connection to a service that is more suitable for the transfer of BCL-PDUs.

Within the ITU, a number of AAL protocols have been identified to adapt the ATM service to a service more suitable for specific types of applications. Two of them can be used to support data communications. These are referred to as AAL 3/4 and AAL 5. We do not come up with a new proposal for an AAL protocol. We assume the use of one of the two protocols recommended by the ITU. For completeness we describe both the AAL 3/4 and the AAL 5 proposal in this section.

First in Section 3.4.1 we will elaborate on the way AAL protocols have been defined within the ITU. In Section 3.4.2 we will describe the service that is provided by the AAL protocols. We will describe the two candidate AAL protocols, AAL 3/4 and AAL 5 in Section 3.4.3 and Section 3.4.4 respectively. We will end this section with a comparison of both protocols in Section 3.4.5.

3.4.1 The ITU View of the AAL

The ITU describes the AAL in its Recommendations I.362 and I.363 ([74], [75]). Five types of AAL protocols have been identified. AAL type 1 has been defined to support constant bit rate services over an ATM network. AAL type 2 has been defined for variable bit rate services that need a constant delay, i.e., that need to

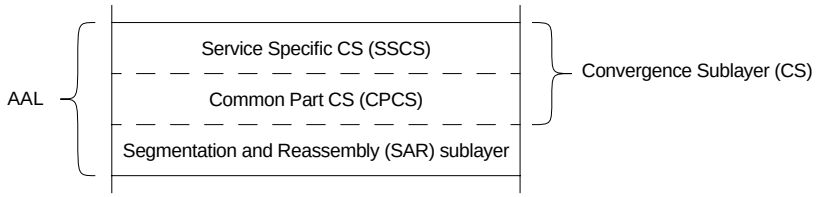


Figure 3-18: Sublayering of the ATM Adaptation Layer

transfer timing information between source and destination. AAL types 3 and 4 are identical and are referred to as AAL 3/4. This protocol has been defined to support variable bit rate services that do not wish to transfer timing information, e.g., data services. A new AAL protocol, AAL 5 ([20]), has been defined, which also supports variable bit rate (data) services. This protocol is supposed to reduce protocol processing and transmission overhead compared to AAL 3/4 ([130]).

Recommendation I.362 specifies the use of AAL 3/4 for the support of connectionless services, but it is also stated that the standardization of other AALs for connectionless services is for further study. From now on we will focus on AAL 3/4 and AAL 5, since these are the protocols that are best suited to support the BCLS.

The AAL has been sublayered in a Segmentation and Reassembly (SAR) sublayer and a convergence sublayer (CS), which has in turn been sublayered into a common part (CPCS), and a service specific part (SSCS) (Figure 3-18). The ITU specifies that for the support of connectionless services, the SSCS will be empty.

For the AAL, only the user plane functionality is considered. Since the AAL is used on an end-to-end basis over an ATM connection, only very little control functionality is needed. An AAL connection corresponds always directly to an ATM connection, although multiplexing of different AAL connections onto a single ATM connection is possible in AAL 3/4.

Since the SSCS will be empty for the support of connectionless services, the service provided by the AAL equals the service provided by the CPCS. In Recommendation I.363, the ITU has defined primitives for the service of the CPCS. However, these primitives form already a refinement of the service primitives that are needed to describe the end-to-end behaviour of the service. The primitives in the recommendation already describe how the local interface could be implemented. It refines the service primitives into a number of interactions at the local interface. At first, it suffices to describe the service at the highest level of abstraction, i.e., without describing how the service primitives can be refined. Later on, we will elaborate on the refinement into interface primitives. The AAL

3/4 and the AAL 5 protocols provide almost the same service to their users. The AAL 5 proposal specifies a few additional parameters for the service primitives. However, these will not be used by the BCL network layer protocol. In the description of the AAL service, we omit these extra parameters. We describe the service as far as it is required by the BCL NL protocols.

3.4.2 AAL service

The AAL service is a connection-oriented service, which allows for the transfer of AAL-SDUs of any size up to 65 535 octets. The ATM-SDUs can be transferred along a previously established connection between a single source AAL-SAP to one or more destination AAL-SAPs. Like for the ATM service, establishment and release of a connection can be done either on demand or (semi-)permanently using management procedures.

Two modes of operation are identified: assured operations, and non-assured operations. In case of assured operations, every AAL-SDU is delivered with exactly the data content that the user sent (with a very high probability). In case of non-assured operations, AAL-SDUs may be lost or corrupted with a certain probability. As an option, corrupted SDUs may be delivered to the receiving user with a notification that the SDU is corrupted in non-assured mode of operation (corrupted data delivery option). For use in a connectionless environment, the ITU recommends an AAL that supports only the non-assured mode of operation. Furthermore, it recommends not to use the corrupted data delivery option.

The AAL specification is not concerned with the connection establishment phase or connection release phase of the communications. This is considered to be handled in the control plane. In this chapter, we will also limit our scope to the data transfer phase of the communication.

In the data transfer phase, user data can be transferred with the following service primitives:

- AAL-DataRequest, with parameter
  AAL-SDU; and
- AAL-DataIndication, with parameter
  AAL-SDU.

If an AAL connection is available, an AAL-DataRequest primitive at a source CEP in an AAL-SAP will result in an AAL-DataIndication primitive at the destination CEP associated with the source CEP. In case multiple CEPs are associated with

the source CEP, the AAL-DataRequest will result in AAL-DataIndication primitives at all destination CEPs. The AAL-SDU parameter of the indication primitives equals that of the corresponding request primitive(s).

Refinement of Service Primitives

In Recommendation I.363 ([75]) the ITU does not specify the service primitives, as we have specified them here. Instead, two possible refinements of the service primitives are specified. The service primitives are refined into a number of interactions, which we will call interface primitives. The SDU is exchanged across the interface (the implementation of a SAP) in Interface Data Units (IDUs), which are parameters of the interface primitives. The refinement applies to the interactions at a particular local interface, i.e., the implementer of an AAL protocol entity is free in choosing the refinement for that particular entity, as long as the implementation of the higher layer protocol entity uses the same refinement of the service primitives at the interface between the two entities.

The refinements of the AAL service primitives are called message mode, and streaming mode. In the *message mode*, the SDU of a service primitive (AAL-DataRequest or AAL-DataIndication) is exchanged as a single IDU in a single interface primitive. In the *streaming mode* the SDU is exchanged in a number of interface primitives, each exchanging a part of the SDU in its IDU.

Message mode. In the message mode, there is a one-to-one mapping from service primitives to interface primitives. Let us use the ITU terminology for the interface primitives, i.e., we call them invoke and signal, instead of the conventional request and indication for service primitives. In the message mode, the following interface primitives have been identified:

- AAL-UNITDATA-invoke, with parameter
  AAL-IDU
  This interface primitive implements the AAL-DataRequest primitive. The only parameter of the primitive is the IDU, with a maximum length of 65 535 octets.

- AAL-UNITDATA-signal, with parameter
  AAL-IDU
  This interface primitive implements the AAL-DataIndication primitive. It has as its parameter the IDU, which implements the SDU parameter.

Streaming Mode. In the streaming mode the service primitives are refined into series of interface primitives. The AAL-SDU parameter of the service primitive is exchanged in a series of consecutive AAL-IDU parameters of the interface primitives. Extra interface primitive parameters are needed to identify to which SDU an IDU belongs. Furthermore, extra primitives are defined to enable user or service provider to abort the exchange of IDUs for a particular SDU at a certain interface. The following interface primitives can be identified:

- AAL-UNITDATA-invoke, with parameters
  AAL-IDU, More, and Maximum Length
  This interface primitive is used to exchange a part of the SDU across the interface, in the form of the IDU parameter. A series of AAL-UNITDATA-invoke primitives implements a single AAL-DataRequest primitive. Apart from the IDU, the AAL-UNITDATA-invoke primitive has a More parameter that specifies whether the IDU contains the end of an AAL-SDU or not. Furthermore, the interface primitive exchanging the first IDU related to an SDU contains a Maximum Length parameter, indicating an upperbound to the length of the AAL-SDU that is being exchanged by means of IDUs.

- AAL-UNITDATA-signal, with parameters
  AAL-IDU, More, and Maximum Length
  This interface primitive is used to exchange a part of the SDU across the interface, in the form of the IDU parameter. A series of AAL-UNITDATA-signal primitives implements a single AAL-DataIndication primitive. The More parameter is used in the same way as for the AAL-UNITDATA-invoke primitive. Only the AAL-UNITDATA-signal primitive exchanging the first IDU related to an SDU contains a Maximum Length parameter, of which the meaning is the same as for the AAL-UNITDATA-invoke primitive.

- AAL-U-Abort-invoke
  This primitive is issued across the local interface if a user that has already exchanged a number of AAL-IDUs related to a certain AAL-SDU wants to abort the exchange of the SDU.

- AAL-U-Abort-signal
  This primitive is issued to indicate that the remote user has aborted the transfer of the AAL-SDU being received.

- AAL-P-Abort-signal
  This primitive is issued to indicate that the provider is not able to complete the transfer of the AAL-SDU being received. Its use is initiated by the provider.

3.4.3 AAL 3/4

The AAL 3/4 protocol has been specified in ITU Recommendation I.363 ([73]) to provide the service specified in the previous subsection. Its general operation is as follows. Upon receipt of an AAL-DataRequest primitive a CPCS-PDU is constructed, containing the AAL-SDU as payload. This CPCS-PDU is segmented into segments of 44 octets. After that, a series of SAR-PDUs is constructed, each containing one of the segments as payload. These SAR-PDUs are sent to the destination AAL protocol entity as the payload of ATM cells. This entity strips the SAR-PDUs, and reassembles the CPCS-PDU using the obtained segments. Finally, it extracts the AAL-SDU from the CPCS-PDU, and delivers it to the service user with an AAL-DataIndication primitive.

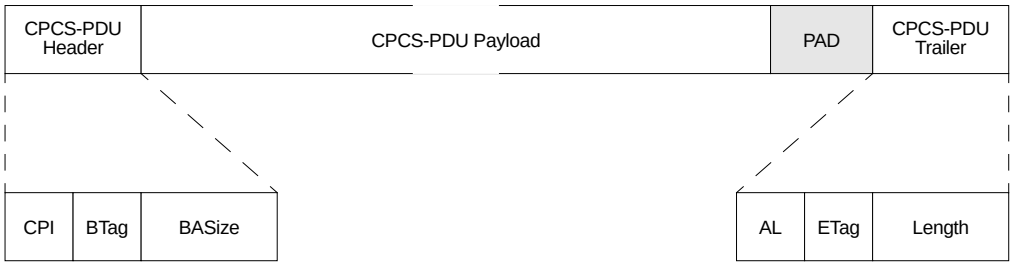
CPCS

Functions performed by the CPCS sublayer are:

- padding of the AAL-SDU to a multiple of 4 octets,
- delimiting of the AAL-SDU by means of a length field in the CPCS-PDU, and
- limited error detection by means of the length field, and by means of (identical) tags in both the header and the trailer of the CPCS-PDU.

Figure 3-19 shows the format of the PDU passed by the CPCS to the SAR sublayer of the AAL 3/4. The following fields can be identified:

- Common Part Indicator (CPI)
  This field is intended to be used for management purposes. Identification of management messages and indication of the counting units for the BASize and Length fields have already been foreseen.
- Begin Tag (BTag) and End Tag (ETag)
  These tags associate the header and the trailer of a CPCS-PDU. The sender inserts the same value in the BTag and the ETag field of a given PDU and changes the value for each successive PDU. For a given PDU, the receiver checks the BTag with the ETag upon receipt, so that the loss or insertion of ATM cells containing a CPCS-PDU header or trailer can be detected.
- Buffer Allocation Size (BASize)
  By means of this field, the sender informs the receiver of the maximum buffer requirements for the receipt of the CPCS-PDU. The BASize is equal to or larger than the length of the Payload field.



CPI	Common Part Indicator	1 octet
BTag	Beginning Tag	1 octet
BASize	Buffer Allocation Size	2 octets
CPCS-PDU Payload		1 ... 65535 octets
PAD	PADding	0 ... 3 octets
AL	ALignment	1 octet
ETag	End Tag	1 octet
Length	Length of CS-PDU Payload	2 octets

Figure 3-19: CPCS-PDU format for AAL 3/4

- CPCS-PDU Payload
  This field contains the AAL-SDU.

- Padding (PAD)
  This field complements the CPCS-PDU Payload field to a multiple of 4 octets.

- Alignment (AL)
  This field is used to complement the length of the CPCS-PDU Trailer to 4 octets.

- Length
  This field is used by the sender to inform the receiver of the length of the CPCS-PDU Payload field. The receiver uses the field to detect loss or gain of information.

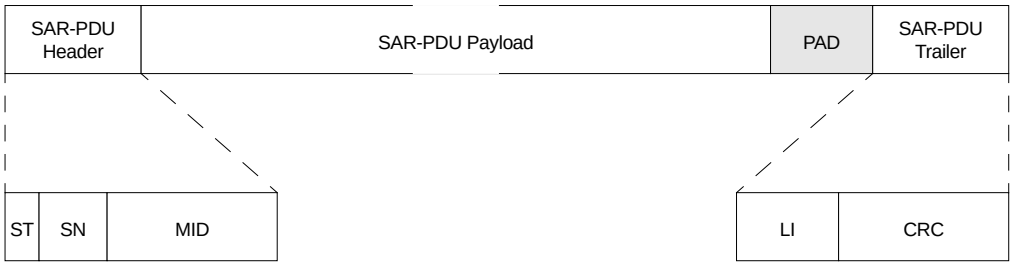
SAR

Functions performed by the SAR are:

- segmentation of CPCS-PDUs into 44 octet segments in the source protocol entity,

- reassembly of the segments into the original CPCS-PDU in the destination entity,

- delimiting of CPCS-PDUs by means of the sequence type and the sequence number field, and by means of the length indication,



ST	Sequence Type	2 bits
SN	Sequence Number	4 bits
MID	Multiplexing IDentification	10 bits
SAR-PDU Payload		1 ... 44 octets
PAD	PADding	0 ... 43 octets
LI	Length Indication	6 bits
CRC	Cyclic Redundancy Check	10 bits

Figure 3-20: SAR-PDU format for AAL 3/4

- error detection by means of the sequence type, sequence number and the CRC field, and
- multiplexing and demultiplexing of different AAL connections by means of the MID field.

Figure 3-20 shows the format of the SAR-PDUs that are passed to the ATM layer. The following fields have been defined:

- Sequence Type (ST)
  This field identifies the SAR-PDU Payload as being either the first (Beginning Of Message, BOM), a middle (Continuation Of Message, COM), or the last (End Of Message, EOM) segment of a CPCS-PDU, or containing a complete CPCS-PDU (Single Segment Message, SSM).
- Sequence Number (SN)
  The source assigns successive numbers (modulo 16) to this field, for the transfer of successive segments of the same CPCS-PDU. These SNs can be checked by the destination in order to detect missequencing, loss, duplication, or misinsertion of segments.
- Multiplexing Identification (MID)
  This field is used to identify SAR-PDUs in case that either SAR-PDUs for different AAL connections are transferred on the same ATM connection, or if SAR-PDUs for different CPCS-PDUs are transferred interleaved.
- SAR-PDU Payload
  This field contains a segment of the CPCS-PDU.

- Padding (PAD)
  If the last or only segment of a CPCS-PDU has a length smaller than 44 octets, this field is used so that the total length of the segment and the PAD field is 44 octets.[3]

- Length Indication (LI)
  This field indicates the length of the SAR-PDU Payload field in octets.

- Cyclic Redundancy Check (CRC)
  This field contains an error detecting code, calculated over the entire SAR-PDU.

Special values of the fields have been defined for the case where the sending user wants to abort the transfer of a CPCS-PDU, e.g., because the AAL-U-Abort-invoke interface primitive has been issued in streaming mode. This so-called Abort-SAR-PDU has its Segment Type set to EOM, and its Length field to 63. The Payload field may be set to 0, and is ignored by the receiver, just like the contents of previously received segments of the CPCS-PDU, currently being transferred.

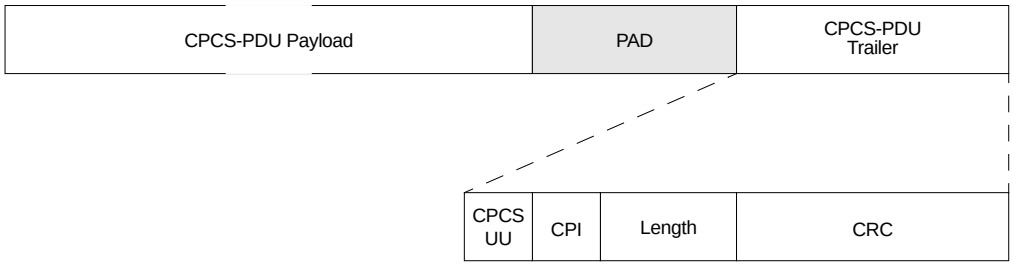
3.4.4 AAL 5

The AAL 5 protocol has been proposed to alleviate the supposed overhead of the AAL 3/4 protocol ([20]). Almost all functionality of the protocol resides in the CPCS sublayer. The only function of the SAR sublayer is the segmentation and reassembly of messages. The SAR does not use PCI to perform this function, instead it uses the ATM User-to-User Information parameter of the ATM-DataRequest and -Indication primitives to detect the last segment of a message.[4] The general operation of the protocol is as follows. Upon receipt of an AAL-DataRequest primitive, a CPCS-PDU is constructed, containing the AAL-SDU as payload. This PDU is segmented into 48 octet segments that are transferred to the destination AAL protocol entity in ATM cells. For the last segment of the PDU, the ATM User-to-User Information parameter is set. This parameter is transferred transparently with the ATM cell to the destination, which is now able to complete the reassembly of the CPCS-PDU. The receiver can extract the original AAL-SDU from the CPCS-PDU, and deliver it to the service user by means of an AAL-DataIndication primitive.

---

[3] In Recommendation I.363 this field is considered part of the SAR-PDU Payload field. In our opinion it is conceptually better to consider it as a separate field.

[4] This ATM User-to-User Information parameter is in fact directly mapped onto the ATM Payload Type field.



CS-PDU Payload		1 ... 65535 octets
PAD	PADding	0 ... 47 octets
CPCS-UU	CPCS User-to-User indication	1 octet
CPI	Common Part Indicator	1 octet
Length	Length of CS-PDU Payload	2 octets
CRC	Cyclic Redundancy Check	4 octets

Figure 3-21: CPCS-PDU format for AAL 5

CPCS

The functions carried out by the CPCS of the AAL 5 protocol are:

- padding of the CPCS-PDU to a multiple of 48 octets,
- delimiting of the AAL-SDU by means of a Length field in the CPCS-PDU trailer, and
- error detection by means of the Length field and a CRC.

Figure 3-21 shows the format of the CPCS-PDU, which is passed by the CPCS sublayer to the SAR sublayer for segmentation. The following fields can be identified:

- CPCS-PDU Payload
  This field contains the AAL SDU.
- Padding (PAD)
  The Padding field complements the CPCS-PDU Payload field, so that the total length of the CPCS-PDU is a multiple of 48 octets.
- CPCS User-to-User indication (CPCS-UU)
  This field is intended to be transparently transferred between the users of the CPCS.
- Common Part Indicator (CPI)
  The use of this field has not been specified yet. It has been defined to align the

CPCS-PDU trailer to a 8 octet boundary. A possible function is the identification of PDUs for layer management.

- Length
  The Length field is used to indicate the number of octets the CPCS-PDU Payload field consists of. It can be used by the receiving entity to identify the end of the payload, and to detect any loss or gain of information (e.g., because of loss or misinsertion of segments).

- Cyclic Redundancy Check (CRC)
  This field contains an error detecting code, calculated over the entire CPCS-PDU.

In order to abort the transfer of a partially transmitted AAL-SDU, e.g., in case the AAL-U-Abort-invoke interface primitive has been issued, the Length field is set to 0.

SAR

The SAR-PDU of the AAL 5 does not contain any PCI. It only contains one 48 octet segment of the CPCS-PDU as its payload (Figure 3-22). The SAR performs the segmentation and reassembly functions. The transfer of the last segment of a CPCS-PDU is signalled by the ATM User-to-User indication parameter.

Figure 3-22: SAR-PDU format for AAL 5

3.4.5 Comparison

Both AAL 3/4 and AAL 5 provide a service that can be used by BCL network layer protocols, and both make use of the service provided by ATM. However, there are quite a number of differences between the two protocols. These are differences in the type of PDUs functions operate on, the functions performed, and the amount of PCI that is generated.

Type of PDUs functions operate on

Protocols for the AAL differ in the way the functions are performed. Functions can either be performed per CPCS-PDU (per packet) or per SAR-PDU (per segment). Table 3-3 classifies the AAL 3/4 and AAL 5 proposals according to the

type of PDUs functions operate on. The segmentation and reassembly functions are not listed in this table, because they do not operate on either segments or packets. They provide for the transformation between segments, which form the payload of ATM cells, and packets.

In general, PCI needs to be exchanged by the communicating protocol entities, in order to be able to carry out a function. If a function is performed on a per segment basis, PCI will also be added to each segment, i.e., to the SAR-PDU. For a function, performed per packet, PCI will be added to the total packet, i.e., to the CPCS-PDU. It will be placed in the header or trailer, and thus be transferred with the first or last segment of the packet. Therefore, performing a function on a per segment basis, tends to result in a larger overhead than performing a function on a per packet basis.

This seems to lead to the conclusion that all functions which need to exchange PCI should be carried out per packet (as it is done in the AAL 5 proposal). However, there are arguments against this conclusion. If the AAL is used in a BCL Network, user data proceeds from source to destination along a number of CLSs interconnected by ATM connections. In order to avoid the need for reassembly and segmentation in each CLS, it seems advantageous to perform the functions that need to be carried out in a CLS on a per segment basis. This issue will be further discussed in Section 4.3. Furthermore, if error detection is performed per segment, the loss or corruption of segments can be detected earlier than if it is performed per packet. In the latter case, loss or corruption is only detected upon receipt of the last segment. For a CLS, this could imply that all other segments of the packet have already been forwarded to the next CLS. If error detection is performed per segment, forwarding of the segments after the corrupted or lost one can be avoided, which is more efficient.

As it can be observed from Table 3-3, AAL 3/4 performs a lot of functions per segment, while AAL 5 performs most functions per packet. This is reflected in the distribution of functionality over the SAR and CPCS sublayers. In principle, the SAR sublayer performs per-segment functions. The CPCS sublayer performs per-packet functions. The AAL 3/4 has most of its functionality in the SAR sublayer, while the AAL 5 has an almost empty SAR sublayer, performing most functions in the CS sublayer.

Functions

There is one significant difference in the functions performed by AAL 3/4 and AAL 5. The AAL 3/4 protocol performs a multiplexing function, which is not

function	AAL 3/4	AAL 5
padding	per segment + per packet	per packet
AAL-SDU delimiting	per segment + per packet	per packet
error detection	per segment + per packet	per packet
multiplexing	per segment	not provided
abort	per segment	per packet

Table 3-3: Classification of AAL proposals

performed by the AAL 5 protocol. In a connectionless environment, a multiplexing function enables a protocol entity to transfer several AAL-SDUs simultaneously. The SAR-PDUs carrying the SDUs are transmitted interleaved on the outgoing ATM connection. This is especially useful if the interface to the protocol operates in streaming mode. Then, the AAL-IDUs constituting an SDU can be passed to the AAL interleaved with the IDUs of other SDUs. AAL 3/4 is able to perform multiplexing because it can identify SAR-PDUs by means of the MID field. The importance of this possibility will be discussed in Section 4.3.

Generated PCI

Figure 3-23 gives an example of the segmentation of an AAL-SDU by the AAL 3/4 protocol. First the CPCS header and trailer are added to the user data, next the obtained CPCS-PDU is segmented into 44-octet chunks. SAR headers and trailers are added to these segments, in order to enable reassembly, and detection of transmission errors. The constructed SAR-PDUs are passed to the ATM layer as ATM-SDUs. These are transmitted by the ATM layer, after adding an ATM header. Padding is used to complement the CPCS-PDU to a multiple of 4 octets, and to complement the last SAR-PDU to a 48-octet boundary.

The total number of octets, $L_{3/4}$, passed to the ATM layer as ATM-SDUs, for the transfer of a single AAL-SDU with a length of N octets is given by

$$L_{3/4} = 48 \times \lceil (N + 8) / 44 \rceil . \qquad (3.1)$$

This can be explained as follows. Each ATM-SDU contains 48 octets. The number of ATM-SDUs to be transferred, equals the number of 44-octet segments that are needed to carry the AAL-SDU plus 8 octets of header and trailer.

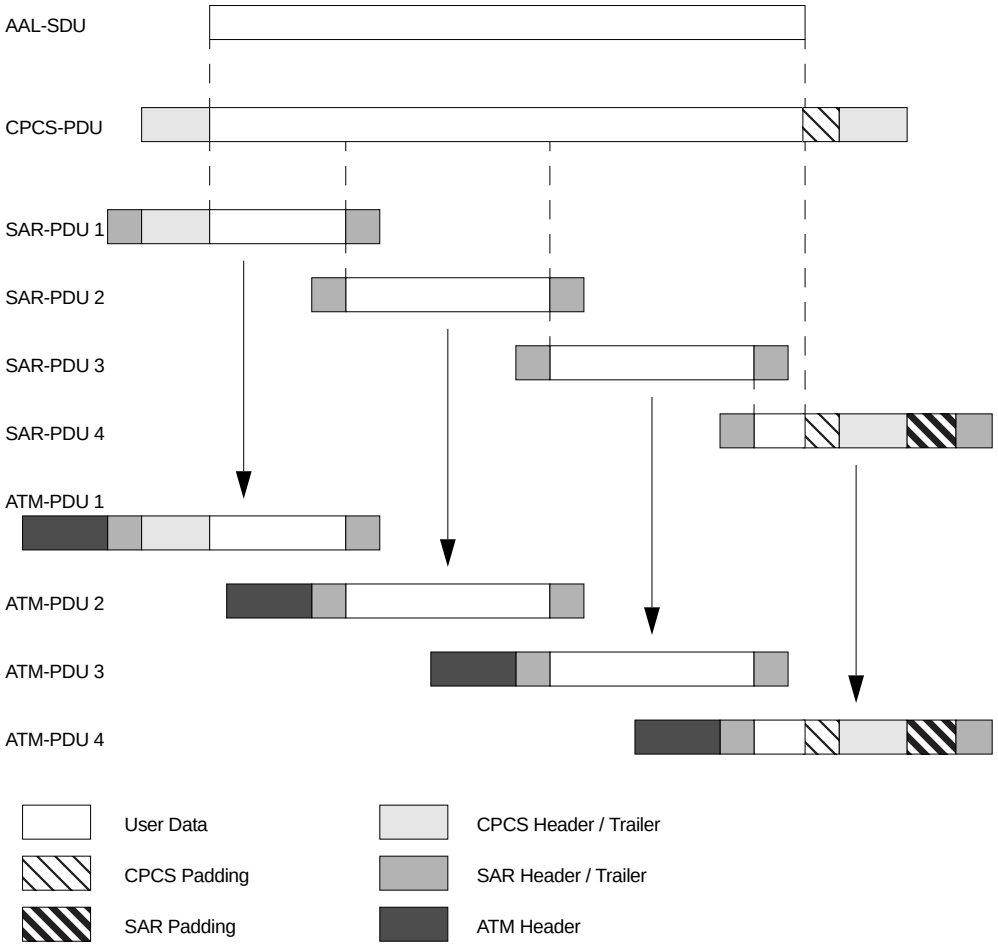


Figure 3-23: Segmentation in AAL 3/4

Figure 3-24 illustrates the segmentation of an AAL-SDU by the AAL 5 protocol. First, CPCS Padding and a CPCS trailer is added to the SDU, so that the resulting CPCS-PDU consists of a multiple of 48 octets. Next, the CPCS-PDU is segmented into 48 octet chunks, which are transferred as the payload of ATM cells.

For the AAL 5, the total number of octets passed to the ATM layer for the transfer of a single AAL-SDU, L_5, is given by

$$L_5 = 48 \times \lceil (N + 8) / 48 \rceil . \qquad (3.2)$$

Each ATM-SDU to be transferred contains 48 octets. The total number of octets is the product of 48 and the number of 48-octet segments needed to carry the AAL-SDU and the 8 octets of header and trailer information.

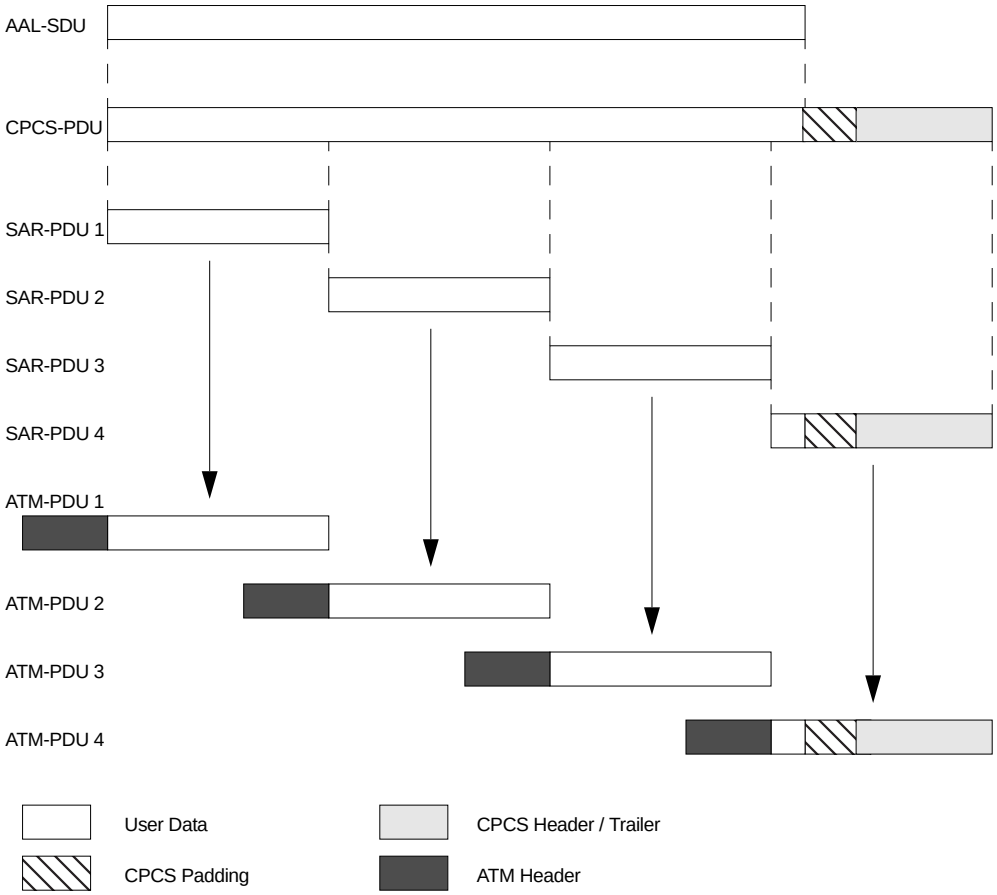


Figure 3-24: Segmentation in AAL 5

In order to show the difference in efficiency between the two AAL proposals, Figure 3-25 shows the relation between N, the AAL-SDU length, and ΔL, the proportional difference in efficiency between AAL 3/4 and AAL 5. This difference has been defined as follows:

$$\Delta L = 100\frac{L_{3/4} - L_5}{L_{3/4}} . \tag{3.3}$$

For large N, ΔL converges to $100 \times (1 - 44/48)$. This means that the AAL 5 protocol needs approximately 8.3% less bandwidth than the AAL 3/4 protocol for the transfer of large packets.

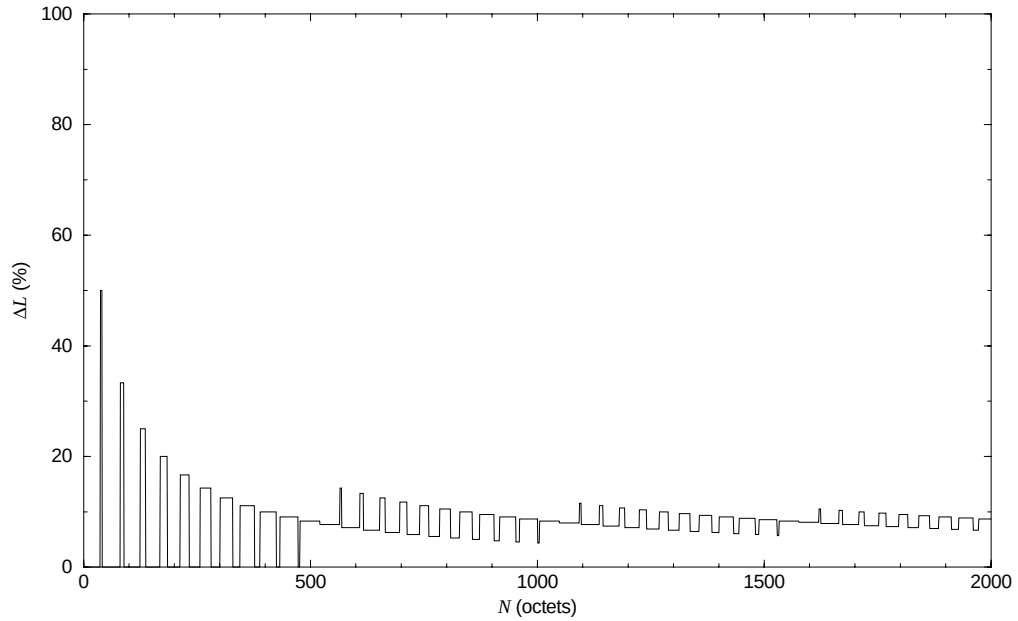


Figure 3-25: Relative Difference in Efficiency between AAL 3/4 and AAL 5

3.5 Network Layer Protocols

The key function of the BCL Network Layer is the routing of PDUs containing the users' BCL-SDUs from source to destination. Additional functions to be performed include address validation, address screening, access class enforcement, and connection management. The BCL Network Layer should perform the conversion from the connectionless BCLS to the connection-oriented AAL service.

In the network architecture for the direct method, we have defined two types of network layer protocols, the BCL Network Access Protocol, and the BCL Network Protocol. The first one is intended for the access of an end-system to the BCL Network. It should take care of maintaining connectivity between the user and the network, address validation, address screening, access class enforcement, and delivery of a PDU containing the users BCL-SDU from the source to the proper access CLS, or from an access CLS to the proper destination. The latter should maintain the connectivity between CLSs in the network, and route the PDU from the CLS connected to the source to the CLS connected to the destination.

Network layer protocols have been defined in different communities. The ITU is currently defining two protocols for the provision of BCDS on B-ISDN ([76], [81]). The Connectionless Network Access Protocol (CLNAP) is intended to be used between an end-system and a CLS. The Connectionless Network Interface Protocol (CLNIP) should be used between CLSs. Similar protocols have been defined to provide SMDS over an ATM-based access network ([6], [125]). These are called SMDS Interface Protocol for Connectionless Service (SIP-CLS), and the Inter-exchange Carrier Interface Protocol for Connectionless Service (ICIP-CLS) respectively.

Because of the similarity of the ITU and SMDS protocols, we will only describe the former ones. We first describe the CLNAP in Section 3.5.1. In Section 3.5.2 we describe the CLNIP.

3.5.1 CLNAP

The Connectionless Network Access Protocol (CLNAP) is being defined in ITU Recommendation I.364 ([76], [81]). Its general operation is as follows. Upon receipt of a BCL-DataRequest, it constructs a CLNAP-PDU. An AAL connection to the proper access CLS in the BCL Network is selected, and an AAL-DataRequest is issued at the proper CEP. In the CLS, upon receipt of the AAL-DataIndication containing the PDU, source address validation, access class enforcement, and destination address screening is performed, and the addresses are considered by the routing function. In the access CLS on the receiver side, the proper AAL connection to an end-system is selected, and source address screening is performed. The CLNAP-PDU is transferred over the connection by issuing an AAL-DataRequest primitive at the corresponding CEP. Upon receipt of the PDU in the end-system as the SDU of an AAL-DataIndication primitive, the BCL-SDU is extracted from the PDU and delivered to the service user by means of a BCL-DataIndication primitive.

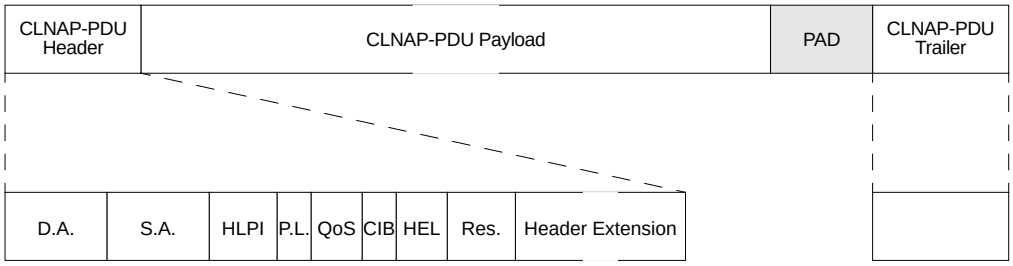
Functions performed by the CLNAP are:

- delimiting of the CLNAP-SDU,
- addressing, in order to allow for the selection of the destination and indication of the source,
- carrier selection, to allow the user to explicitly select a preferred connection on a per SDU basis,
- QoS selection, to allow for selection of the desired QoS on a per SDU basis,

- padding of the SDU to a multiple of 4 octets, and optionally
- error detection by means of a CRC.

Figure 3-26 shows the format of the PDU passed by the CLNAP to the AAL. The following fields can be identified:

- Destination Address
  This field indicates to which protocol entity(ies) the CLNAP-PDU is destined.
- Source Address
  This field indicates which protocol entity originated the CLNAP-PDU.
- Higher Layer Protocol Indicator (HLPI)
  This field identifies to which protocol entity using the service of the CLNAP the SDU must be delivered.
- PAD Length
  This field indicates the length of the PAD field in octets.
- Quality of Service (QoS)
  This field indicates the QoS requested by the user for the transfer of the SDU.
- CRC Indication Bit (CIB)
  This field indicates the presence of a CRC field.



D.A.	Destination Address	4 octets
S.A.	Source Address	4 octets
HLPI	Higher Layer Protocol Identifier	6 bits
P.L.	PAD Length	2 bits
QoS	Quality of Service	4 bits
CIB	CRC Indication Bit	1 bit
HEL	Header Extension Length	3 bits
Res.	Reserved	2 octets
Header Extension		0 ... 20 octets
CLNAP-PDU Payload		1 ... 9188 octets
PAD	PADding	0 ... 3 octets
CRC	Cyclic Redundancy Check	4 octets (optional)

Figure 3-26: CLNAP-PDU format

- Header Extension Length (HEL)
  This field indicates the length of the Header Extension field in 32-bit words.
- Reserved
- Header Extension
  The use of this field has not been defined yet in [76]. It can for instance be used for the indication of the required carrier.
- CLNAP-PDU Payload
  This field contains the BCL-SDU.
- Padding (PAD)
  This field complements the CLNAP-PDU Payload to a multiple of 4 octets.
- Cyclic Redundancy Check (CRC)
  This field is optional. If it is present it contains the result of a CRC32 calculation performed over the entire CLNAP-PDU.

3.5.2 CLNIP

The Connectionless Network Interface Protocol (CLNIP) is also being defined in ITU Recommendation I.364 ([76], [81]). It is intended for use between CLSs. Its PDU format is only slightly different from the CLNAP PDU format. An Access CLS converts a received CLNAP-PDU into a CLNIP-PDU, and forwards this one to the next CLS, based on the address information of the PDU. Transit CLSs just have to forward the PDUs. In the Access CLS, connected to the destination of the PDU, the CLNIP-PDU is converted back into an CLNAP PDU, and forwarded to the relevant end-system.

The functions to be performed for the CLNIP are the same as those to be performed for the CLNAP, except for the carrier selection function, which is not performed. Furthermore, an additional function called encapsulation has been defined, which may be applied in Access CLSs.

Encapsulation is a technique where an entire CLNAP-PDU, including its header, is transferred in the payload field of a CLNIP-PDU. In the Access CLS, connected to the destination, the CLNAP-PDU is extracted again. This is called decapsulation. If encapsulation is not applied, only the payload of the CLNAP-PDU is transferred by the CLNIP-PDU. Its header is replaced by an CLNIP-PDU header. These headers may be identical.

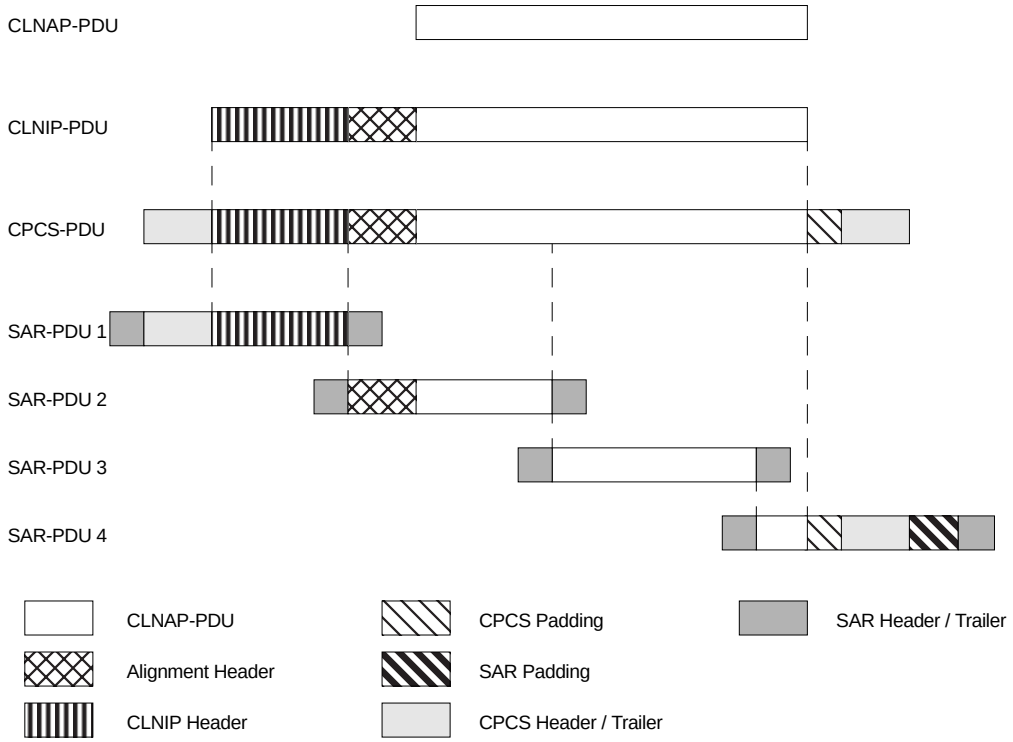


Figure 3-27: Encapsulation of a CLNAP-PDU

The PDU format of the CLNIP is identical to the one for the CLNAP (Figure 3-26) if encapsulation is not applied. If encapsulation is applied, the optional CRC field is not used. Furthermore, the header is extended with a Header Extension Post-Pad. This field pads the Header Extension field to a total of 20 octets, so that the total length of the CLNIP-PDU header is 40 octets. If AAL 3/4 is used to transfer the PDU, this header plus the CPCS header (4 octets) fit exactly in a single SAR-PDU Payload field (44 octets). Thus, in case of encapsulation an extra ATM cell, containing the new CLNIP-PDU header, is transferred before the cells containing the CLNAP-PDU are forwarded. An extra 4 octet alignment header is inserted before the CLNAP-PDU to fill the space that was originally occupied by the CPCS-PDU header, which is now positioned before the CLNIP-PDU (see Figure 3-27).

3.6 Summary and Concluding Remarks

In this chapter, we have presented an architectural framework for the provision of a connectionless service with an ATM-based B-ISDN. It is obtained using a method of functional decomposition of a service provider into a number of

protocol entities cooperating according to the rules of a protocol, and an underlying service, which they use for their communication.

As a starting point, abstract descriptions of the service to be provided, i.e., the Broadband Connectionless Service (BCLS), and the service of the underlying system, i.e., the ATM service, have been given. These have been derived from available documents concerning SMDS, and from ITU recommendations.

We have introduced two candidate network architectures, in which different protocols and protocol entities are identified. The network architectures are defined by recursively decomposing the BCLS until the ATM service is obtained as the underlying service. The first architecture assumes that only functionality up to the ATM layer is present in the network. Conversion of the connection-oriented ATM service to a connectionless service is done on an end-to-end basis. This is called the indirect method of providing a connectionless service. The second network architecture is based on the direct method. It assumes additional functionality in the network, which converts from the connection-oriented ATM to connectionless communications. According to this architecture, special nodes, called Connectionless Servers (CLSs) are installed in the network, to perform this conversion by routing packets from incoming to outgoing ATM connections, based on the address information contained in the packet. The latter network architecture is the most suitable one for situations where individual end-systems need to transfer packets to a lot of different destinations, since it reduces the number of needed ATM connections.

A protocol reference model has been derived from the network architectures. It identifies two layers on top of the ATM layer, which are needed to provide the BCLS. These are the ATM Adaptation Layer and the Broadband Connectionless Network Layer.

Finally, we have overviewed protocols that can be used at the different layers, identified in the network architecture. These protocols are the AAL 3/4 protocol and the AAL 5 protocol for the AAL, and the CLNAP and the CLNIP for the BCL Network Layer. They are all being standardized within the ITU.

It can be concluded that the installation of CLSs within the B-ISDN is the most important extension that is needed to let it provide a connectionless service. Besides Physical and ATM layer functions, these CLSs need to perform protocol functions from the AAL and the BCL Network Layer. Chapters 4, 5, and 6 of this dissertation focus on the design and analysis of a CLS.

They say that choice is freedom.
I'm so free it's driving me insane.

Joe Jackson - Laughter and Lust

Chapter 4
Implementation Aspects of Connectionless Servers

In this chapter, we focus on the direct method of providing a connectionless service. We study the implementation of the CLSs, which are responsible for routing data packets through the ATM network. In the network architecture, presented in the previous chapter, we have identified the functional (protocol) entities that constitute a CLS. In this chapter, we study the problem how a real system can implement these functional entities. We present a number of candidate implementation architectures, which describe how the functionality of the entities to be implemented can be divided over modules of a CLS. These modules are the constituting parts of a real system, e.g., processor boards. For the purpose of designing an implementation architecture from the network architecture, we transform the functional entities and their interactions to modules and their interconnection (see Figure 4-1).

We discuss the implementation of a CLS in more detail with respect to a number of important issues. These are the reassembly of network layer packets, the copying and the buffering of data within the CLS, and the integration of a CLS within an ATM switch. Furthermore, we analyse the proposed implementation architectures with respect to some of the design criteria given in Chapter 1, i.e., availability and scalability.

First, in Section 4.1, we present the various implementation architectures that can be used to implement a CLS. Then, in Section 4.2, we describe the functionality of the identified modules in more detail. In Section 4.3, we address a number of implementation issues that heavily influence the ultimate design, and have a strong impact on the performance of the system. Finally, in Section 4.4, the evaluation of the implementation architectures according to a number of criteria will be discussed.

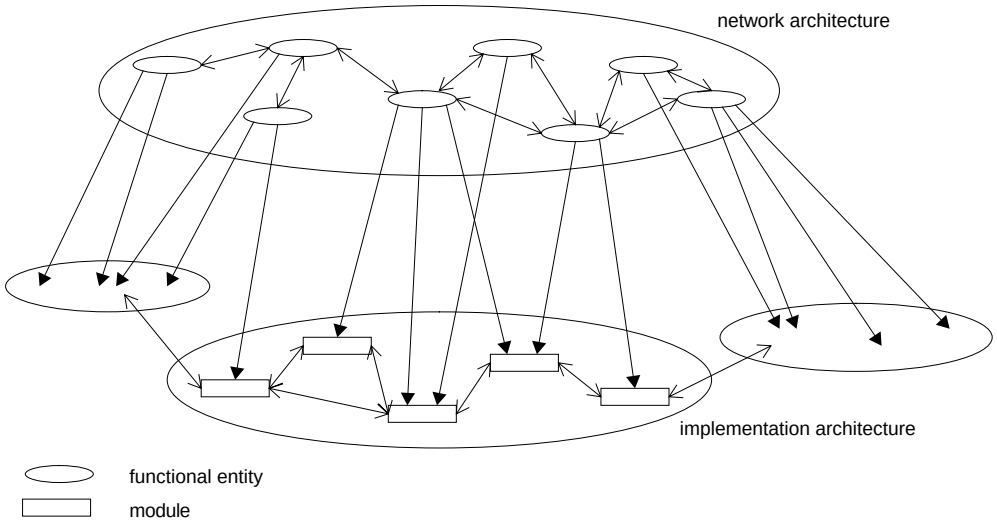


Figure 4-1: Design of an Implementation Architecture

4.1 Implementation Architectures

Once the functionality of a CLS has been well specified, a first step towards its implementation is to design an implementation architecture. In the implementation architecture, we determine which modules will constitute the functionality, the functionality of the individual modules, the interactions between the modules, and the way they are physically interconnected. In this section, we identify six different implementation architectures, starting from a basic one, and ending with a complex one aiming at high performance, which can easily be integrated in an ATM switch.

First, in Section 4.1.1, we discuss the implementation architecture of an ATM switch, and recall the functional entities that should be implemented in a CLS. Next, in Sections 4.1.2 through 4.1.7, we present the six different CLS implementation architectures.

4.1.1 General

Before discussing the implementation of a CLS, let us spend some time on the implementation of an ATM switch. This is useful since a CLS and an ATM switch have much in common. Both should route ATM cells that arrive on an incoming link to the proper outgoing link, and update some of the protocol control information contained in the cells. Furthermore, one might want to integrate a CLS in an ATM switch for a number of reasons, which will be discussed in Section 4.3.4.

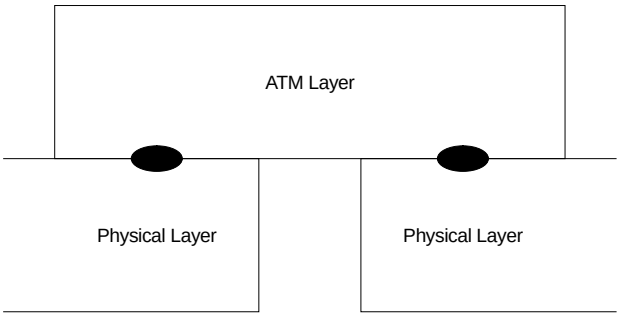


Figure 4-2: Functional Model of an ATM Switch

In Chapter 3, we have already identified an ATM switch (or node) as a real system that will be needed to provide the BCLS. Figure 4-2 shows a functional model of an ATM switch, which has been derived from part of the model in Figure 3-4. A number of protocol entities, performing functions residing in specific layers of the BCL Protocol Reference Model, can be identified in the functional model. Besides Physical Layer processing, the switch has to perform the switching function of the ATM layer.

In most implementations ([140]) functionality is distributed over a number of modules (see Figure 4-3). Figure 4-3 describes the same functionality as Figure 4-2. However, whereas the latter shows an abstraction of the system, with boundaries and SAPs between abstract subsystems (i.e., protocol entities), the former shows the mapping of these abstract subsystems onto real subsystems, i.e., modules. In Figure 4-3, different modules are represented by different shadings. Points of interaction between modules are denoted as open ovals. The order in which the different modules operate on an arriving data unit is indicated by an arrow.

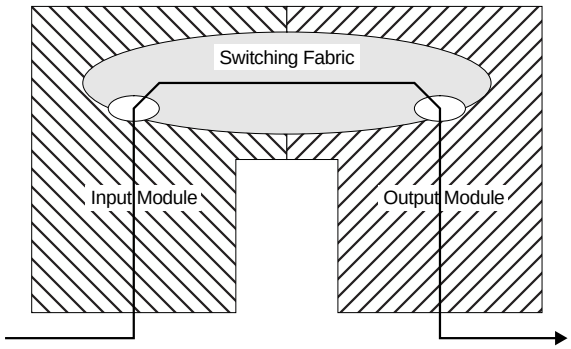


Figure 4-3: Distribution of Functionality in an ATM Switch

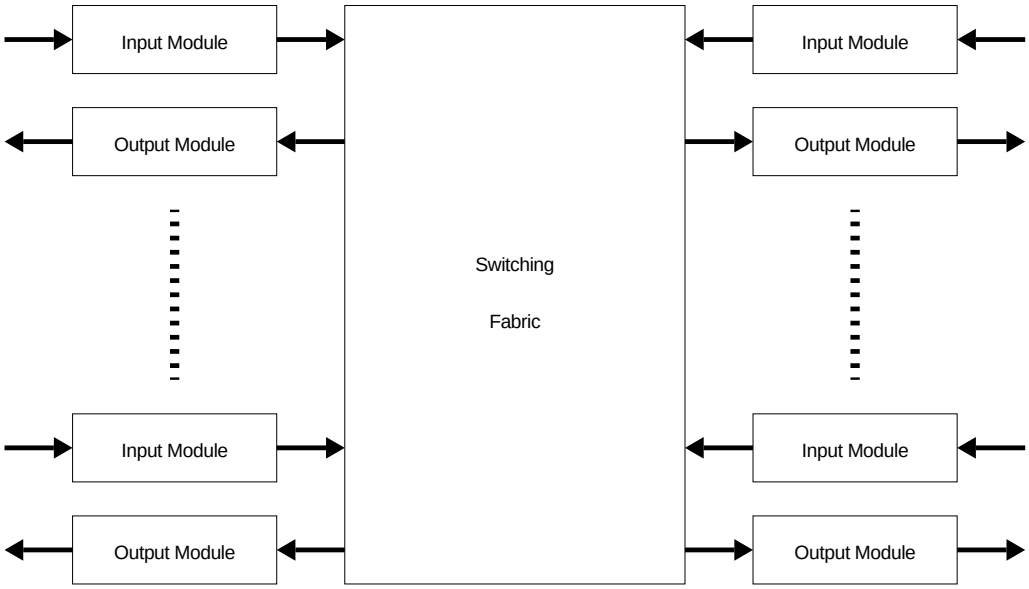


Figure 4-4: Architecture of an ATM Switch

Three types of modules can be identified in an ATM switch. The main module is the Switching Fabric. Its function is to forward ATM cells from the Input Modules to the proper Output Modules. The format of the switch data units, i.e., the unit of information the Switching Fabric operates on, can but does not necessarily have to be the same as the format of the ATM cell. The path a switch data unit has to take through the Switching Fabric is determined by a routing tag, i.e., a header field of the switch data unit, and information present in the fabric. An Input Module performs Physical Layer functions, and ATM header processing. It will convert ATM cells to switch data units, e.g., by segmentation. Furthermore, it will determine a routing tag for the switch data unit. An Output Module will convert switch data units to ATM cells, e.g., by removing the routing tag and possibly by reassembling a number of switch data units. It will perform ATM and Physical Layer processing, needed to submit the cell over the outgoing ATM interface. In general, it is possible to shift some of the functionality between Input Module, Switching Fabric and Output Module.

According to this distribution of functionality, a switch architecture as shown in Figure 4-4 is obtained. A large number of Input and Output Modules is connected to a single Switching Fabric.

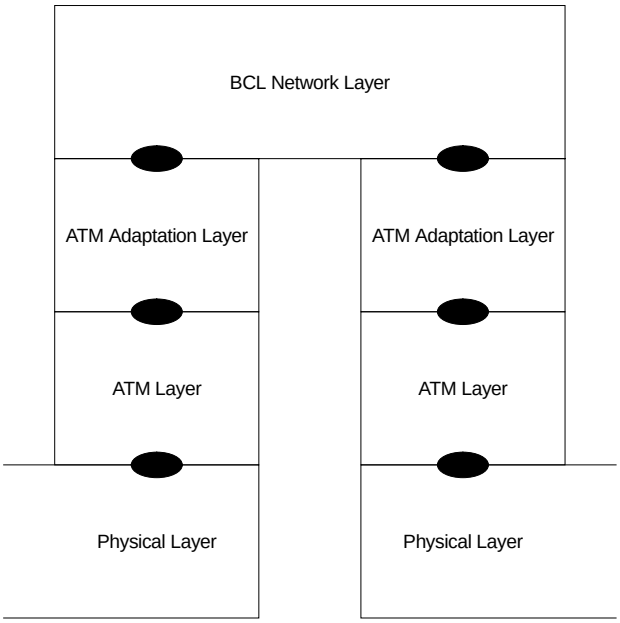


Figure 4-5: Functional Model of a CLS

A functional model of a CLS is shown in Figure 4-5. Switching is now performed at the BCL Network Layer. Furthermore, the CLS has to perform processing for the Physical Layer, the ATM Layer, the ATM Adaptation Layer, and the BCL Network Layer.

In general, a CLS will also have Input and Output Modules, which perform Physical Layer and ATM Layer functions, just like in an ATM switch ([105]). These can be very similar to the ones described above for an ATM switch. Additionally, modules performing functions of the AAL and BCL Network Layer can be found in a CLS. These will be called Connectionless Service Modules (CLSMs). In general, a CLS connects to multiple input links, hence an Interconnection Structure (IS) is needed as well. This IS forwards data units between the different modules in the CLS. An example of such an IS is an ATM Switching Fabric.

Additional to the modules that will be identified below, a CLS will contain one or more modules that perform management tasks. Functions of such a module are among others to download and update tables in other modules, and to detect failures of other modules. Such a management module will communicate with the other modules in the CLS either using the common IS, other via a dedicated interconnection structure. We do not take the management modules into further consideration.

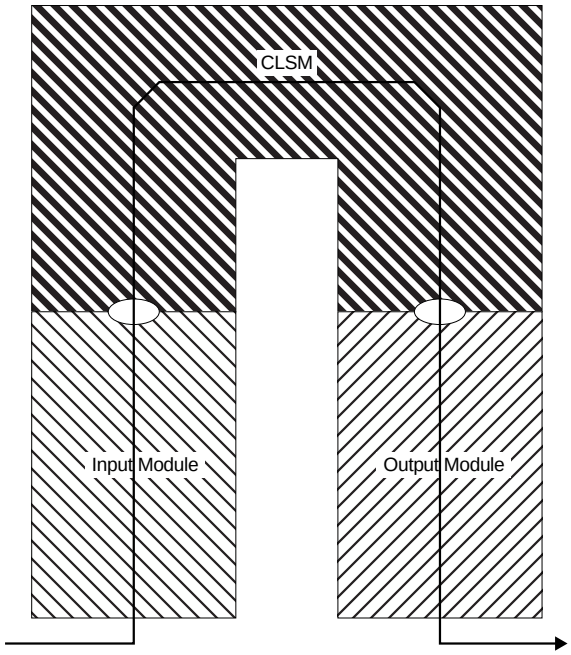


Figure 4-6: Distribution of Functionality (Architecture 1)

In the following subsections, a number of different implementation architectures will be presented. They differ in the modules that are identified, and in the way these modules are interconnected, i.e., the transformation from network architecture to implementation architecture has been performed differently.

4.1.2 Architecture 1

According to Architecture 1, a CLS consists of a single CLSM. No IS is used. The CLS terminates only a single ATM link (Figure 4-6). The Input and Output Modules for this link are connected directly to a CLSM. The CLSM performs all the processing of the AAL and BCL Network Layer.

ATM cells received by an Input Module are processed, and their payload and an internal connection identifier are forwarded to the CLSM. From the incoming cell stream, the CLSM (virtually[1]) reassembles the packet. A routing function, implemented in the CLSM determines the proper outgoing ATM connection, based on the address information contained in the packet. Next, the CLSM (virtually) segments the packet into cell payloads and forwards them to the Output Module

[1] In Section 4.3.1, it will be discussed that processing and forwarding of the segments of a BCL-PDU can be performed on the fly. This will be referred to as virtual reassembly, as opposed to 'real' reassembly. With (virtual) reassembly we refer to both.

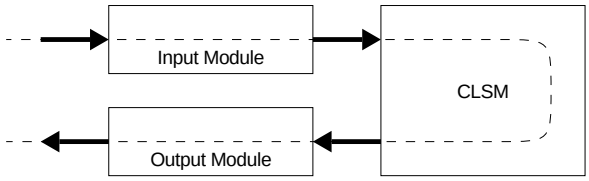


Figure 4-7: Implementation Architecture 1

together with an identifier for the required outgoing ATM connection. This module generates a cell header and transmits the ATM cell on the outgoing link.

We obtain an architecture consisting of a single Input Module, a single CLSM, and a single Output Module (Figure 4-7). In this figure, the dashed line indicates the path of (the segment of) a BCL-PDU.

This implementation architecture is the most basic one that can be identified. The architecture can perform all functions that a CLS should perform. However, it allows only for small capacity CLSs, which are connected to only a single ATM link, since no switching is performed. We now identify a second alternative, that allows for CLSs of a larger capacity.

4.1.3 Architecture 2

According to Implementation Architecture 2, a CLS consists of several CLSMs interconnected by an IS. We distinguish between Input CLSMs, performing processing for incoming packets and routing, and Output CLSMs performing the processing for outgoing packets. Each Input or Output CLSM is connected to an Input or Output Module respectively (Figure 4-8).

ATM cells received by an Input Module are processed, and their payload is forwarded to the corresponding Input CLSM, together with an internal connection identifier. From the incoming cell stream, the Input CLSM (virtually) reassembles the packet. A routing function, implemented in the Input CLSM determines the proper outgoing ATM connection and link, based on the address information contained in the packet. The packet and an indication of the required outgoing connection are forwarded via the IS to the Output CLSM that is responsible for the outgoing link. The packet is forwarded in a format suitable for the IS, i.e., in switch data units. This could be data units with the same length as a single segment. The Output CLSM (virtually) segments the packet into cells, and forwards them to the connected Output Module (possibly interleaved with other packets). This module generates a cell header and transmits the ATM cell on the outgoing link.

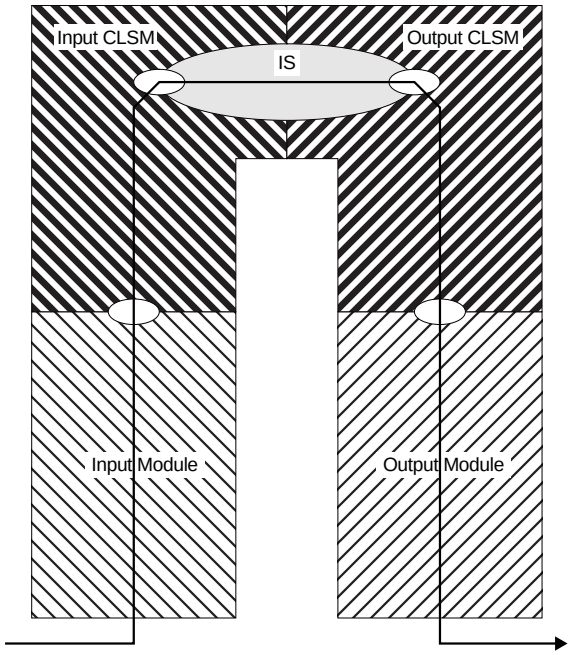


Figure 4-8: Distribution of Functionality (Architecture 2)

Using this distribution of functionality, an implementation architecture is obtained with one Input Module, one Input CLSM, one Output CLSM and one Output Module per ATM link (Figure 4-9). The IS can be such that it switches segments or whole packets, depending on the operation of the CLSMs.

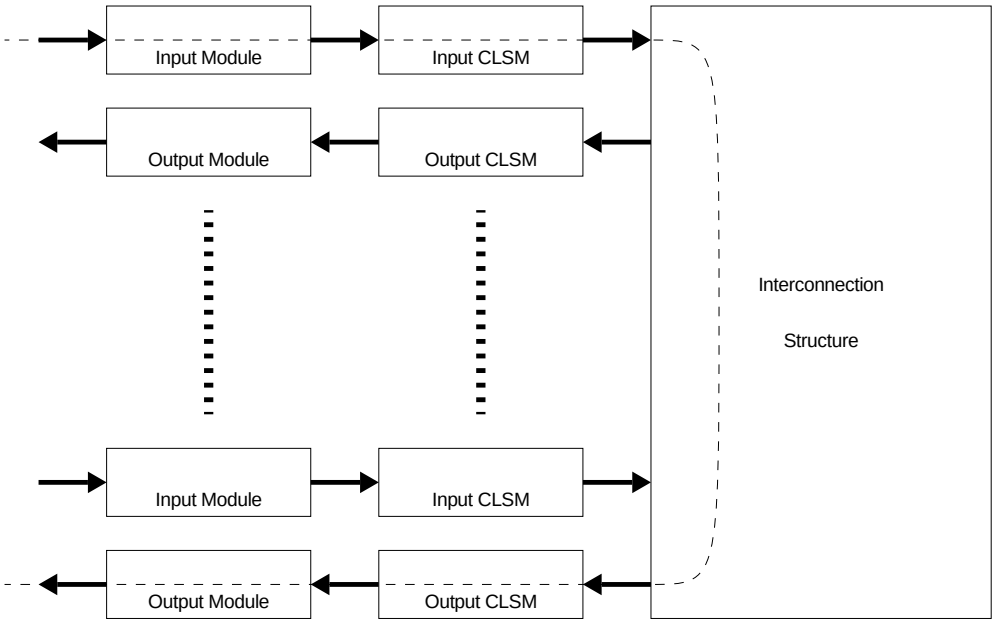


Figure 4-9: Implementation Architecture 2

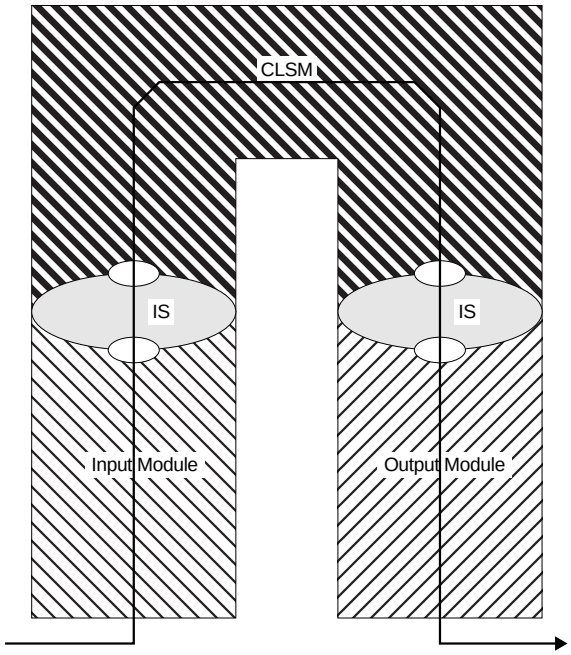


Figure 4-10: Distribution of Functionality (Architecture 3)

Using this architecture, a CLS can be scaled until the throughput of the IS is exceeded. Each ATM link has its own dedicated CLSM (Input as well as Output). A possible bottleneck is the throughput of a single CLSM, which restricts the throughput of a link. In order to relieve this problem, another implementation architecture is introduced, which uncouples the Input and Output Modules from the CLSMs, thus allowing for independent scalability of the CLSMs.

4.1.4 Architecture 3

According to Architecture 3, Input Modules, Output Modules, and CLSMs are all grouped around a central IS. No fixed relationship between an Input or Output Module and a CLSM exists. An Input Module forwards a cell to one of the CLSMs, depending on the ATM connection the cell belongs to, i.e., depending on the VPI / VCI (Figure 4-10).

ATM cells received by an Input Module are processed and their payload and an internal connection identifier are forwarded via the IS to a CLSM. To which CLSM the cell is forwarded depends on the incoming ATM connection (VPC / VCC) on which the cell arrives. An ATM connection is terminated in a single CLSM, which maintains state information for that connection. This CLSM (virtually) reassembles a packet from the incoming cells. A routing function determines the proper

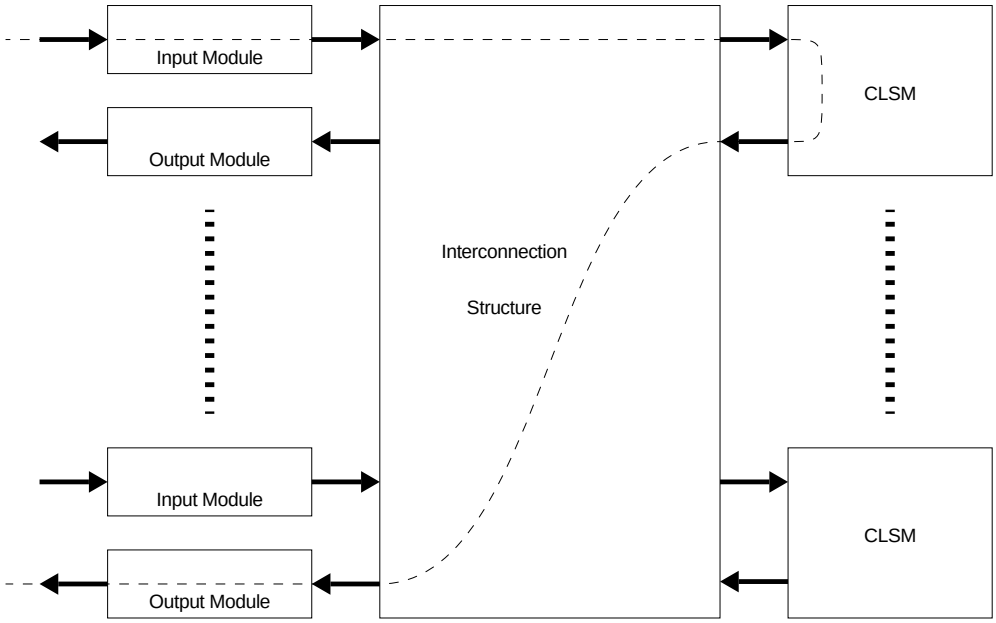


Figure 4-11: Implementation Architecture 3

outgoing link and outgoing ATM connection for the packet, using the address information contained in the packet. The packet is (virtually) segmented again, and the segments are forwarded to the Output Module, responsible for the outgoing link, via the IS. The Output Module constructs an ATM cell, which is transmitted on the outgoing link.

The architecture corresponding to this distribution of functionality does not enforce a one-to-one mapping between Input or Output Modules and Input respectively Output CLSMs (Figure 4-11). One Input and one Output Module is needed per ATM link. The number of CLSMs used in a CLS can depend on the expected traffic load.

This implementation architecture allows for a flexible dimensioning of the CLS. The Input and Output Modules should be dimensioned in such a way that they can cope with the maximum cell rate on a link. Furthermore, the total number of CLSMs should be such that the CLS can achieve the required packet throughput with the required QoS. A disadvantage of this architecture is that all data is switched by the IS twice. Another disadvantage is that different CLSMs may have to send cells to the same outgoing ATM connection, because the cells belong to packets that are destined for the same CLS. As a result, coordination between CLSMs is required for traffic shaping and multiplexing on that connection, or several connections to the same CLS should be maintained. We introduce another

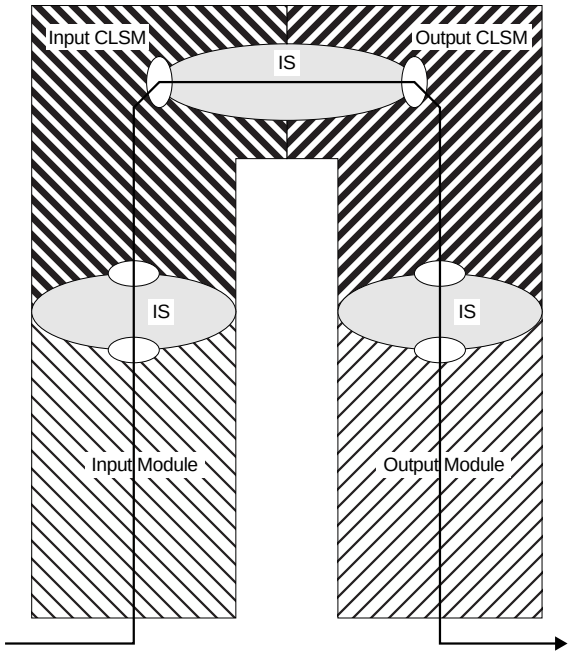


Figure 4-12: Distribution of Functionality (Architecture 4)

implementation architecture, which avoids the need for coordination or maintaining several connections.

4.1.5 Architecture 4

According to Architecture 4, the functionality of the AAL and BCL Network Layer is implemented in separate modules for input and output, i.e., Input CLSMs and Output CLSMs. Input Modules, Output Modules, Input CLSMs, and Output CLSMs are all connected to a central IS (Figure 4-12). Each incoming ATM connection is handled by a single Input CLSM, and analogously each outgoing ATM connection is handled by a single Output CLSM.

ATM cells are processed by the Input Module where they first arrive. Based on the incoming ATM connection, the payloads are forwarded, via the IS to the proper Input CLSM. After (virtual) reassembly and network layer processing, an outgoing link and outgoing ATM connection are determined by the routing function of the Input CLSM. The packet and an indication of the required outgoing connection are forwarded via the IS to the Output CLSM responsible for the outgoing ATM connection. This Output CLSM maintains state information for each outgoing connection it handles. It performs the necessary network layer processing, and (virtually) segments the packet, and forwards it to the Output

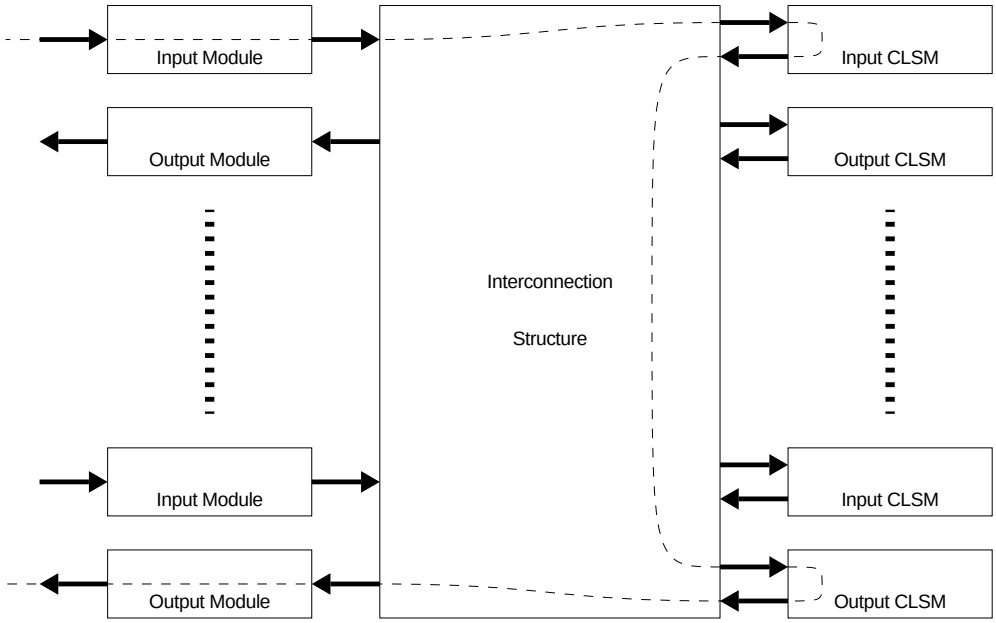


Figure 4-13: Implementation Architecture 4

Module that is connected to the proper outgoing link. This Output Module generates new ATM headers, and transmits the cells on the ATM link.

This distribution of functionality allows, like Implementation Architecture 3, a flexible mapping of ATM links (and Input and Output Modules) to Input and Output CLSMs. Furthermore, it allows, like Architecture 2, for each CLS or end-system that is connected to the system, to cooperate with a single Input CLSM and a single Output CLSM. Figure 4-13 shows the architecture that will result from the distribution.

This implementation architecture seems to combine the advantages of architectures 2 and 3. However, it has one important disadvantage. All data is switched by the IS three times. In order to solve this problem, another implementation architecture is identified.

4.1.6 Architecture 5

According to Architecture 5, all AAL and BCL Network Layer functions for a packet are performed in a single CLSM, except for some specific functions, which are performed in a separate module called Output Packet Processing Module (PPM). These are functions that maintain state information, which is specific for the next node, a packet will be forwarded to. By performing these functions in the same Output PPM for all the packets that should be forwarded to that next node,

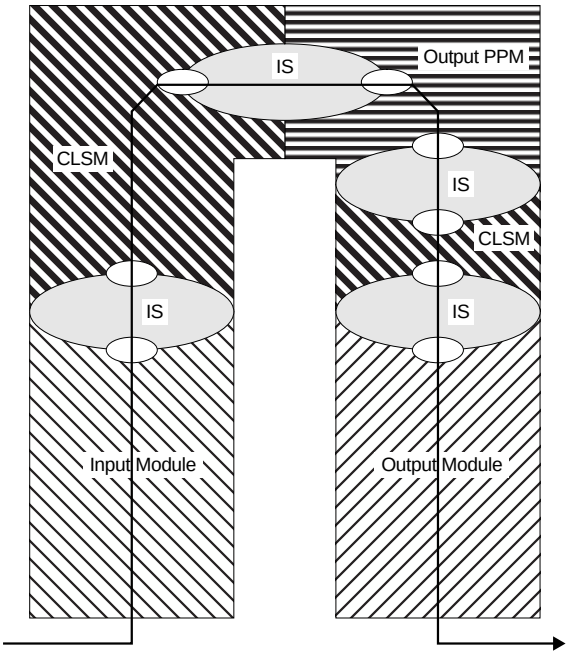


Figure 4-14: Distribution of Functionality (Architecture 5)

it can be avoided that several copies of the information have to be maintained in different modules. An example of such a function is source address screening. By performing this function always in the same Output PPM, it can be avoided that copies of the list of allowed source addresses for a certain destination end-system have to maintained in several modules. For its operation, an Output PPM only needs the information contained in the first cell of a packet, and the result can be returned to the CLSM in a single switch data unit. Input Modules, Output Modules, CLSMs, and Output PPMs are all connected to a central IS (Figure 4-14).

ATM cells are processed by an Input Module upon arrival. Depending on the incoming ATM connection (VPI/VCI) their payload is forwarded to a certain CLSM. This CLSM maintains state information for all connections it terminates. Here, AAL processing takes place, and a packet is (virtually) reassembled. Network layer processing for incoming packets takes place, and routing is performed. After the routing decision has been taken, the network layer header, the CPCS header, and an indication of the required outgoing connection are forwarded to a specific Output PPM, depending on the required outgoing ATM connection. This Output PPM maintains state information for all ATM connections that originate there. In the module, processing for outgoing packets and possibly AAL multiplexing is performed. The result, e.g., multiplexing information (an MID value) is returned to the CLSM in a single switch data unit. Upon

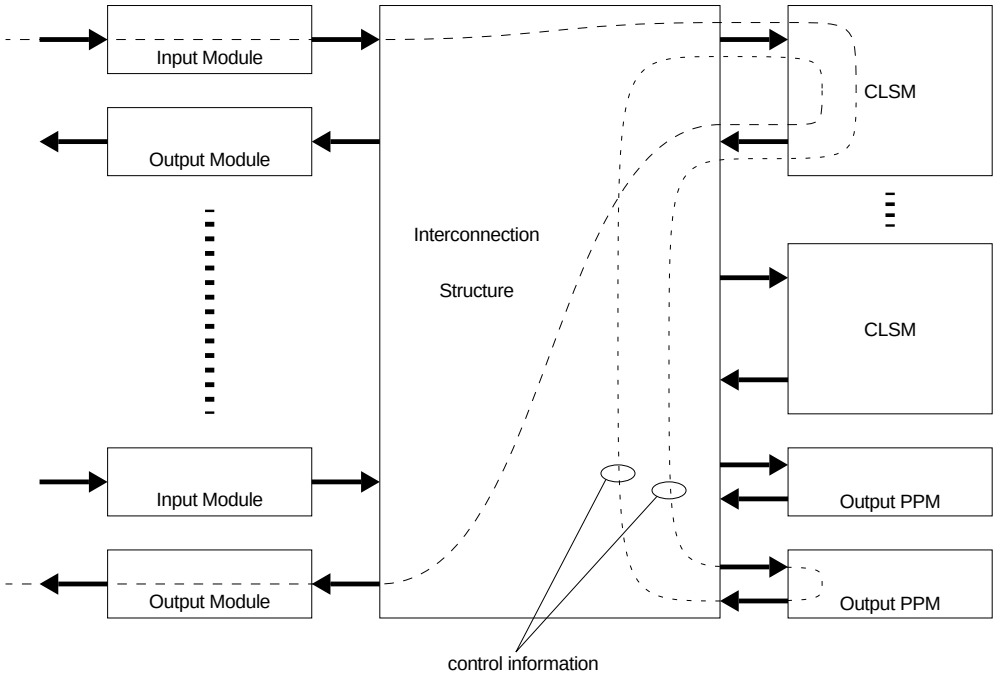


Figure 4-15: Implementation Architecture 5

receipt of this result, the CLSM (virtually) segments the packet again, performs additional AAL layer processing, and forwards the cells via the IS to the appropriate Output Module. This module performs ATM layer processing, and sends the cells out on the outgoing link.

Figure 4-15 shows the implementation architecture for this distribution of functionality. In this architecture, there is a flexible mapping between Input and Output Modules and CLSMs. Furthermore, all the output processing for a single end-system can be performed in the same Output PPM. However, since only header information is forwarded from the CLSM to this module, unnecessary switching of data can be avoided.

4.1.7 Architecture 6

Routing is a very demanding function that has to be performed in a CLS. It involves looking up the destination address in a possibly large routing table. It is a potential performance bottleneck of a CLS. Therefore, it may be wise to remove the one-to-one mapping of CLSM to routing function from the implementation architecture, in order to remove the bottleneck. This leads to the following implementation architecture.

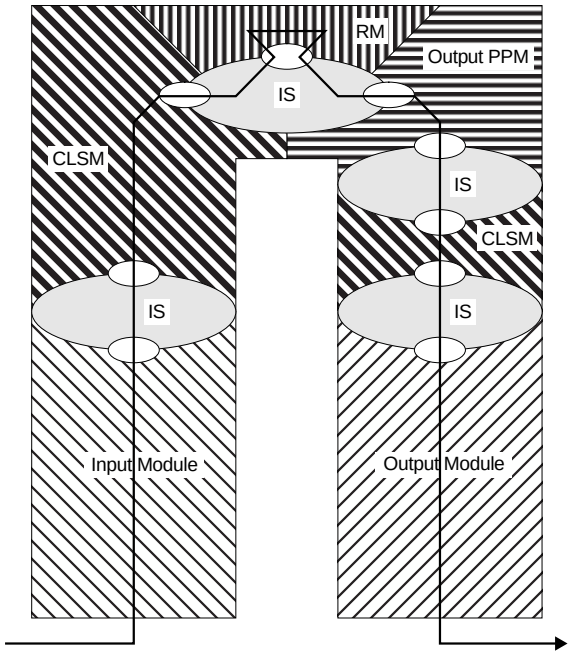


Figure 4-16: Distribution of Functionality (Architecture 6)

According to Architecture 6, routing is performed in separate Routing Modules (RMs), which are connected to the IS. Furthermore, Input and Output Modules, CLSMs, and Output PPMs are connected to the central IS (Figure 4-16).

Incoming ATM cells are processed by an Input Module, and their payload is forwarded to one of the Input CLSMs, depending on the incoming ATM connection. Here, a packet is (virtually) reassembled from the incoming cell stream. After (or during) performing network layer input processing, the network layer header and the CPCS header are forwarded to one of the RMs. The decision, which RM to forward the information to is taken randomly, or based on the load of the RMs. No state information related to ATM connections is maintained here. This module determines the outgoing ATM connection and link, based on the network layer destination address. This information, together with the received information, is now forwarded to the Output PPM responsible for the outgoing ATM connection. After performing additional network layer processing for the output, a switch data unit with an identifier for the outgoing ATM connection and possibly multiplexing information is returned to the CLSM. The CLSM (virtually) segments the packet again, performs AAL processing, and forwards the cells of the packet to the Output Module responsible for the outgoing link. This module constructs the ATM cell header, and transmits the ATM cell on the outgoing link (Figure 4-17).

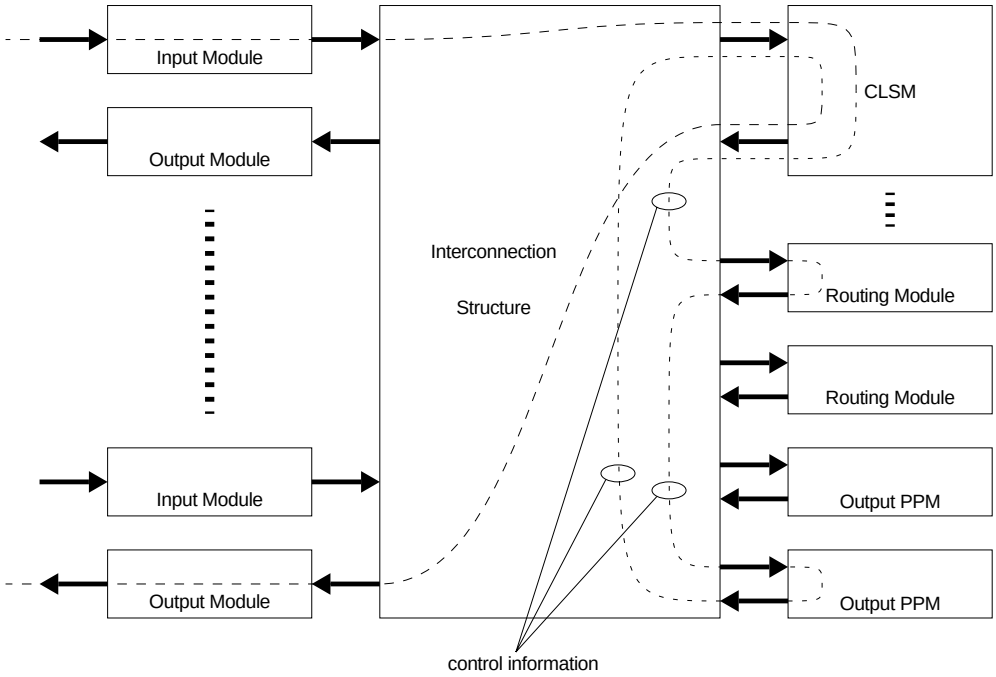


Figure 4-17: Implementation Architecture 6

Like the previous implementation architectures, this one allows a flexible mapping between links and CLSMs, only cells belonging to the same ATM connection on a link have to be forwarded to the same CLSM. Furthermore, it allows a flexible mapping between CLSMs and the routing function, performed in a separate RM. The decision, which RM to forward a packet to can be taken randomly per packet, or depending on the load of the RMs. This modularity of the implementation allows for an optimal balance between the processing capacity for the different functions, i.e., the number of modules performing the same function in parallel can be chosen such that the throughput for each group of parallel modules is as desired.

4.2 Functional Description of the Modules

In the previous subsection, we have identified a number of modules, which will implement the functions that need to be performed in a CLS. In order to provide more insight in the distribution of functionality in a CLS, we will now describe the functions to be performed in each of the modules in more detail.

Depending on the implementation architecture, not all modules are present in every design. If an RM or an Output PPM is not present in an architecture, its

functionality is assumed to be present in CLSMs. Furthermore, we describe the functionality of both an Input and an Output CLSM, while we often do not distinguish between the two. In that case, the CLSM is supposed to have the functionality of both.

First, the IS is described in Section 4.2.1. The Input and Output Modules are described in Section 4.2.2 and Section 4.2.3 respectively. The CLSM is described in Section 4.2.4. Its functionality can be split into an Input CLSM and an Output CLSM, which will be described in Section 4.2.5 and Section 4.2.6. Alternatively, part of its functions could be performed by an Output PPM and possibly a RM. We end with their descriptions in Section 4.2.7 and Section 4.2.8 respectively.

4.2.1 Interconnection Structure (IS)

The sole purpose of the IS is to move information between modules. The main functions that can be identified in the IS are switching and buffering.

Interactions between the IS and other modules constitute of exchanging switch data units. A switch data unit consists of control information and payload. A data unit contains at least an indication of the required destination module when it enters the IS. When it leaves the IS, it contains at least an indication of the source module. The size of the payload is such that it can contain the parameters of an ATM-DataRequest or ATM-DataIndication primitive and an internal connection identifier. Alternatively, the IS can be implemented in such a way that a number of switch data units need to be exchanged for the transfer of the parameters of a single primitive.

4.2.2 Input Module

The Input Modules of a CLS have the same functions as the Input Modules of an ATM switch. They connect to the links to other nodes in the network. The functionality consists of Physical Layer functions and ATM functions. Furthermore, if the Input Module is connected to an IS, conversion of ATM cells to switch data units is performed.

Physical Layer functions include line termination, cell delineation, and Header Error Control (HEC) field verification. ATM Layer functions can include translation of VPI/VCI, routing tag insertion, rate adaptation, and possibly Usage Parameter Control (UPC).

A management module will provide an Input Module with a VPI/VCI mapping table. In this table, it can look up the VPI/VCI of an incoming ATM cell, and obtain an internal connection identifier and also the routing tag for the CLSM the cell has to be forwarded to.

4.2.3 Output Module

The Output Modules of a CLS have the same functionality as those of an ATM switch. Similarly to an Input Module, an Output Module has to perform Physical Layer functions, ATM Layer functions, and conversion of ATM cells from the IS format.

Physical Layer functions include line termination, HEC field generation, and transmission frame generation. ATM Layer functions can include VPI/VCI generation, rate adaptation, and buffering.

An Output Module maintains a table where it can look up for each internal connection identifier, which VPI/VCI value it has to give to the corresponding field in the ATM cell header. This table will be updated by a management module.

4.2.4 Connectionless Service Module (CLSM)

The functionality of the CLSM, as present in implementation architectures 1 and 2, equals that of both the Input CLSM (Section 4.2.5) and the Output CLSM (Section 4.2.6) as described below. In implementation architectures 5 and 6, the functionality of the Output PPM (Section 4.2.7) and the functionality of both the Output PPM and the RM (Section 4.2.8) is excluded from the CLSM respectively.

4.2.5 Input CLSM

In Section 3.3.4, we have identified two different types of CLSs, Access CLSs, and Transit CLSs. The functions to be performed by an Input CLSM depend on the peer entity it communicates with. If the peer entity is another CLS the Input CLSM has to perform only transit functions. Otherwise, if the peer entity is a system outside of the public part of the network (in case of an Access Server), e.g., an end-system or an IWU, the Input CLSM will have to perform access functions as well as transit functions.

The following transit functions have to be performed by an Input CLSM:

- (Virtual) Packet Reassembly
  The Input CLSM should be able to reconstruct the packet (CLNAP-PDU or CLNIP-PDU). This does not imply that the entire contents of the packet needs to be present in a CLSM at a certain moment in time. However, the Input CLSM, RM, and Output CLSM should be able to consult the PCI of the packet.

- Packet Integrity Verification
  A limited amount of errors due to bit errors, cell los, missequenceing, or insertion can be detected, depending on the type of AAL used. The Input CLSM should detect these errors, and take the appropriate measures (e.g., deletion of entire packets if one of its segments is corrupted or lost).

- The functions listed for the RM (Section 4.2.8)

If needed, the following access functions will be performed by the Input CLSM:

- Address Validation
  The Input CLSM should check the source address specified in the packet against a table of source addresses that is allowed for the particular user. The user can uniquely be identified by looking at the incoming link, and the incoming VPI/VCI.

- Destination Address Screening
  The Input CLSM should compare the destination address of the packet with a table of allowed addresses for the given source address.

- Access Class Enforcement
  The rate at which data is arriving from a user should be checked against the rate corresponding to the agreed access class.

A number of tables are to be maintained in an Input CLSM. A table with allowed source address values for each internal connection identifier is used by the address validation function. A table of allowed destination address values for each source address is used by the destination address screening function. Furthermore, a routing table, containing for each destination address, an identifier for the Output Module and an internal connection identifier is present in the Input CLSM. The tables are downloaded and updated by a management module.

4.2.6 Output CLSM

For the Output CLSM, we can also distinguish between transit functions and access functions. The following transit functions need to be performed:

- Cell Payload Construction
  The Output CLSM should transfer the received packets in segments that fit in ATM cells. In general, this will be the same segments that are received by the Input CLSM, but depending on the AAL type used, some of the AAL PCI has to be modified. If packets are really reassembled in the Input CLSM, also packet segmentation has to be performed.

- Traffic Shaping
  Associated with an outgoing ATM connection, is a traffic descriptor, as part of a traffic contract, which characterizes the traffic that can be expected on the connection. This descriptor is used by the ATM network to reserve the proper resources for a connection. The parameters of this traffic descriptor are also used by the traffic shaping function, to shape the outgoing cell stream on a connection, so that the traffic contract is not violated. Cells that would violate the contract are delayed until they can be safely transmitted on the connection.

- The transit functions listed for the Output PPM (Section 4.2.7)

The access function to be performed in an Output CLSM is the one listed for the Output PPM. The status information to be maintained in the Output CLSM includes that listed for the Output PPM. Furthermore, parameters for the traffic shaping function need to be maintained.

4.2.7 Output Packet Processing Module (PPM)

A transit function that may be performed in an Output PPM is the following:

- Multiplexing (Packet Interleaving).
  In order to allow the interleaving of several packets on an outgoing connection, the Output CLSM can multiplex segments from different packets. For this purpose, the MID field has been defined in the AAL 3/4 protocol. AAL 5 does not allow for the interleaving of several packets on an outgoing VC connection. However, it allows for interleaving on an outgoing VP connection, where the VCI is used for packet identification.

Furthermore, the following access function should be performed:

- Source Address Screening
  The Output CLSM should compare the source address of the packet with a table of allowed addresses for the given destination address.

The state information to be maintained in the Output PPM includes a table of used MIDs (VCIs in case AAL 5 is used over VP connections), which is used by the multiplexing function. This table is updated by the PPM. For the source address screening function, a table of allowed source addresses for each destination address is maintained in the Output PPM. The table is updated by a management module.

4.2.8 Routing Module (RM)

The RM has only a single function:

- Selection of outgoing ATM connection
  Based on the destination address given in a packet, the RM will determine the ATM connection (VP / VC) and link on which the packet has to be transferred.

For this function, a routing table will be maintained, indicating for each destination address which Output Module, and which internal connection identifier should be used. The table will be downloaded and updated by a management module.

4.3 Implementation Issues

Given an implementation architecture for a CLS, a number of important issues need to be addressed before detailed implementation decisions can be made. They will be addressed below. First, in Section 4.3.1, the reassembly of packets in the CLS will be discussed. In Section 4.3.2, it will be discussed how copying of data in the CLS can be avoided. In Section 4.3.3 we will discuss how the buffering of SAR-PDUs can take place. Finally, in Section 4.3.4, we discuss the issue of integrating a CLS in an ATM switch.

4.3.1 Packet Reassembly

An important issue in the design of the CLS is the question whether or not a network layer packet should be completely reassembled in the server. In principle, the BCL Network Layer, which is implemented in the CLS, operates on packets. For that reason, this packet should be reassembled from the SAR-PDUs, which are transferred in ATM cells. Before transmitting the packet on the outgoing link, it needs to be segmented again, in order to be able to fit into the SAR-PDUs, which are to be transferred in ATM cells. This mode of operation,

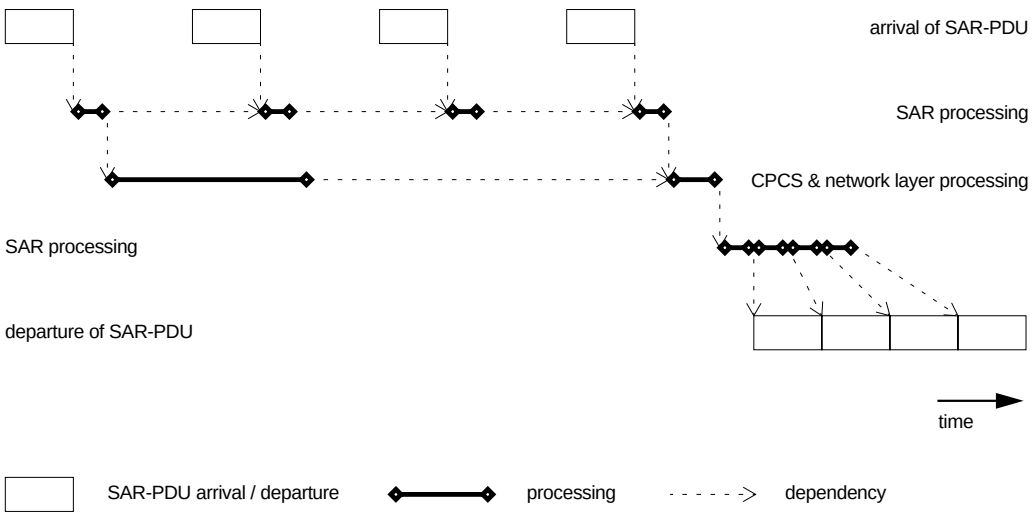


Figure 4-18: Dependencies between Processing in Case of Message Mode

where packets are completely reassembled before transmission, is called *message mode.*

As an alternative to complete reassembly, the packet may be processed on a per-cell basis. Upon receipt, cells are processed one by one, and transmitted again. State information is maintained in the CLS, in order to be able to process subsequent cells of the same packet. This mode of operation is called *streaming mode.* The two modes correspond to the modes of refining the AAL service primitives, discussed in Section 3.4.2.

Reassembly of packets in each intermediate system (e.g., CLS) increases delays, and is inefficient in terms of required buffer capacity. However, there can be a requirement for the CLS to operate in message mode. For specific functions, the processing needs to be completed before the first cell of the packet can be sent out. Such a function may need the presence of the complete packet. If this is the case, message mode is the only suitable mode of operation.

Figure 4-18 shows the dependencies (by means of dashed arrows) between the arrival of SAR-PDUs, completion of the AAL and network layer processing, and the departure of the SAR-PDUs in the case where message mode is required. In general, the CPCS and network layer processing can start as soon as the first cell of a packet has been received, since the CPCS-PDU header and the network layer PCI is contained in this cell[2]. Part of the CPCS processing can only be done after the last SAR-PDU has arrived, and has been processed. This is due to the fact that some of the PCI is contained in this SAR-PDU (it is in the trailer of the CPCS-

PDU). The message mode of operation should be used if SAR processing for the first outgoing SAR-PDU has to wait until completion of all CPCS and network layer processing, as is shown in Figure 4-18. As a result, the first SAR-PDU of a packet can only depart after the last one has arrived. Two possible reasons for this dependency can be identified:

1. Error handling for the incoming packet (i.e., error detection and discarding of erroneous packets) may require the presence of all SAR-PDUs of the packet. In order to avoid partial transmission of erroneous packets, error detection for the entire packet should be completed before header construction and transmission of the first SAR-PDU of a packet can take place. Otherwise, the first couple of SAR-PDUs of a packet could have been sent already at the moment the error is detected. Furthermore, if AAL 5 is used, errors in the address field are only detected after the AAL header analysis for the last SAR-PDU of the packet, since the CRC code for the entire packet is contained in the CPCS-PDU trailer in the last SAR-PDU.

2. In some cases SAR-PDUs of a packet cannot be transmitted interleaved on the outgoing connection with SAR-PDUs belonging to other packets. In this case, a CLS should wait with transmitting a cell until the complete packet it belongs to has been received. Otherwise, delay or loss of one of the subsequent SAR-PDUs of that packet would cause excessive delays to other packets, since their transmission has to wait until the last SAR-PDU of the previous packet has been transmitted. Cells of different packets cannot be interleaved on an outgoing connection if AAL 5 is used over VC connections because AAL 5 does not have a multiplexing (MID) field.

The streaming mode of operation is likely to result in an improved performance, compared to the message mode. Operation in streaming mode is only possible, if the dependency described above does not exist. In this case, AAL header construction and transmission for the first cell of a packet can start as soon as the network layer processing related to the CPCS-PDU header and the network layer has been performed. Figure 4-19 shows the dependencies between the arrival of AAL-PDUs, completion of the AAL and network layer processing, and the departure of the AAL-PDUs in this case. From the figure, it can be seen that the SAR processing for the first outgoing SAR-PDU can start as soon as the CPCS and network layer processing performed upon its arrival has finished. It does not

[2] ITU-T Recommendation I.364 specifies a PDU header of 20 octets with a possible 20 octets extension for the CLNAP ([76]). Together with the 4 octet CPCS-PDU header (in case of AAL 3/4) this makes up at most 44 octets, which fits in the SAR-PDU payload field.

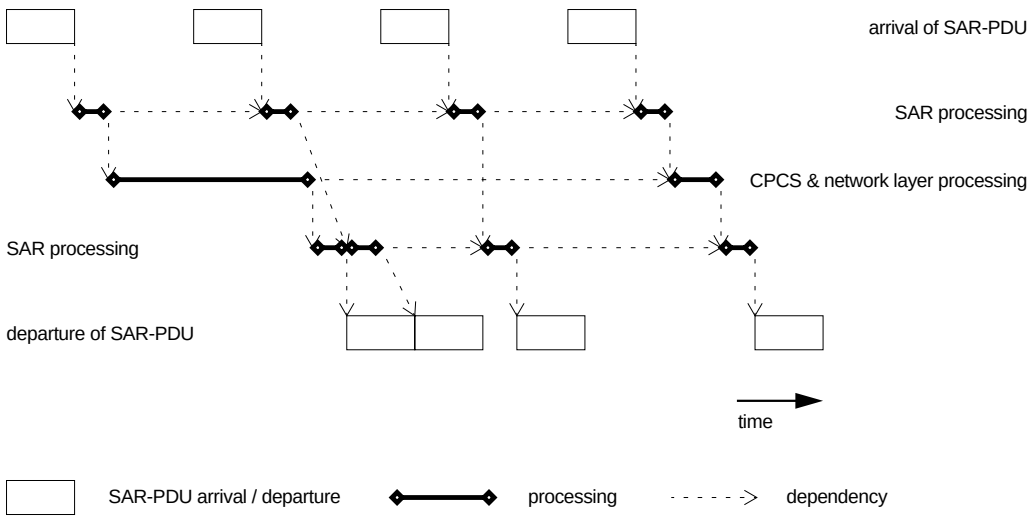


Figure 4-19: Dependencies between Processing in Case of Streaming Mode

have to wait until the CPCS and network layer processing related to the last SAR-PDU of a packet has completed.

In streaming mode, an extra SAR-PDU routing table, containing information about the route of all packets that are being transferred, needs to be maintained. For the first SAR-PDUs of the packet, the outgoing ATM connection, and in case of AAL 3/4, the outgoing MID value is determined, using the main (packet) routing table. This information is stored in the SAR-PDU routing table, in order to be able to use it for consecutive SAR-PDUs of the packet. For SAR-PDUs, which are not the first one of a packet, the outgoing ATM connection and possibly MID is looked up in the SAR-PDU routing table. Furthermore, in case a SAR-PDU is the last one of the packet, the entry for that packet is removed.

Disadvantages of the streaming mode are that it can only function well if interleaving is possible, and that partially corrupted packets are proceeding through the network. It is still an open issue whether the expected performance advantage (see Chapter 5) of the streaming mode is outweighed by the disadvantages.

4.3.2 Data Copying

Most of the functions that need to be performed in a CLS do not need the presence of the user information in the packet. Instead they only need to know some of the PCI of the concerned protocol layer. If one of the modules is able to extract the appropriate information from the incoming cell stream, send it to other modules, which process the PCI, and buffer the cell while waiting for the

result of the operation, it can be avoided that data is copied too often. Furthermore, in this way some of the functions can be performed simultaneously.

Copying of data is undesirable in a CLS. Transferring all cells of a packet from one module to another module involves an additional delay, especially when the cells are transferred via the IS. Furthermore, if all cells of a packet are transferred using the IS several times, the required throughput of the IS increases linearly with the number of times the data is transferred.

The need for data copying depends on the distribution of functionality over the modules in an implementation architecture. If functions that need to process all cells of a packet are concentrated in a few modules, copying can be avoided to a large extent. These modules can extract the information that is needed by other modules (e.g., address information) and transfer it to them. The latter modules in turn need to return only the result of their processing.

Of course Input and Output Modules operate on all the cells, since these are only concerned with cells, not with packets. The cells enter an Input Module on a link from another CLS, or from an end-system. They have to depart from an Output Module on a link to another CLS, or to an end-system. Somehow, these cells need to be transferred from an Input Module to an Output Module. Since additional processing is needed, i.e., AAL and network layer processing, at least one other module should be involved. Architecture 1 employs this basic scenario. However, because of reasons given in Section 4.1, an IS is often needed to interconnect the modules. Furthermore, it turned out to be desirable to connect Input and Output Modules directly to the IS.

Taking the above into account, we still have a number of additional constraints to deal with in determining the distribution of functionality that is most optimal with respect to the amount of data copying. The first one is related to the information contained in the packet (either PCI or payload) and packet related results from other functions that are needed by a certain function. Clearly, both the needed information from the packet and the needed results from other functions should be available in the module where the function is implemented. Table 4-1 shows for each of the AAL and network layer functions to be implemented in the CLS, what information it needs, and what information it results in. Let us consider the source address screening function as an example to illustrate the table. The only information this functions needs of a packet are the source and destination address. This information should be obtained from the (virtual) packet reassembly function. The result of the source address screening is an indi-

function	needed information	resulting information
(virtual) packet reassembly & packet integrity verification	all AAL-PDUs + incoming ATM connection	packet (incl. address information)
address validation	source address + incoming ATM connection	ok / not ok
destination address screening	source address + destination address	ok / not ok
access class enforcement	source address + packet length	ok / not ok
selection of outgoing ATM connection	destination address	outgoing ATM connection
source address screening	source address + destination address	ok / not ok
multiplexing	outgoing ATM connection	multiplexing information
cell payload construction	packet + ok / not ok multiplexing information	AAL-PDUs
traffic shaping	number of AAL-PDUs	departure times for AAL-PDUs

Table 4-1: Needed and Resulting Packet Related Information for Functions

cation whether or not the packet should be forwarded (ok / not ok). This information will be used by the cell payload construction function, which should only process a packet if the result of the address screening is “ok”.

Avoiding unneeded copying of data implies that it should be avoided that the complete packet is needed in several modules. Therefore, it would be wise to implement the (virtual) packet reassembly and packet integrity verification and the cell payload construction in a single module. This is exactly what has been proposed in implementation architectures 1, 3, 5, and 6.

The second additional constraint comes from the fact that certain information has to be shared for the processing of different packets. This information can be static, e.g., the allowed source addresses for a destination address, which are used for source address screening. It can also be dynamic, i.e., it is updated during the processing of a certain packet. This is for instance the case with traffic shaping, where traffic statistics have to be maintained for an outgoing connection, which have to be updated for each packet. Table 4-2 shows for each of the AAL and network layer functions, which shared information it needs for its processing, which shared information it results in, and which packets’ processing operates on the same shared information. Let us consider the source address screening

function	needed shared information	resulting shared information	shared by all packets with the same
(virtual) packet reassembly & packet integrity verification	multiplexing and reassembly info	multiplexing and reassembly info	incoming ATM connection
address validation	allowed source addresses	-	incoming ATM connection
destination address screening	allowed destination addresses	-	source address
access class enforcement	access class + traffic characteristics	traffic statistics	source address
selection of outgoing ATM connection	outgoing ATM connection	-	destination address
source address screening	allowed source addresses	-	destination address
multiplexing	multiplexing info	multiplexing info	outgoing ATM connection
cell payload construction	-	-	
traffic shaping	traffic statistics	traffic statistics	outgoing ATM connection

Table 4-2: Needed and Resulting Shared Information for Functions

function again. The function needs to have a table of allowed source addresses, which is used for the processing of all packets with the same destination.

In order to avoid maintaining several copies of shared information, it would be best to implement all functions using the same shared information in a single module. As an example, it would be best to implement source address screening for a specific destination in a single module. This is not always possible, mainly because of scalability reasons. A more severe constraint comes from the sharing of dynamic information between packets. Implementing functions operating on the same information in different modules would require complex synchronization between the modules in order to let each module have the most up-to-date information. This would for instance be necessary if traffic shaping for the same outgoing ATM connection would be performed in several modules. Since this synchronization will be very hard to realize, an outgoing ATM connection should be used by only a single (Output) CLSM. This implies all traffic destined for a certain CLS or end-system should depart from the same CLSM, or that several ATM connections to this CLS or end-system should be maintained. The former

option is only possible in Architectures 2 and 4 (and in Architecture 1, which has only a single CLSM). The other implementation architectures should maintain several outgoing connections to the same CLS or end-system.

It can be concluded that there is often a trade-off between avoiding data copying, and designing a flexible large-scale implementation. It is still an open issue which architecture has the best overall suitability. In fact, the choice for a specific implementation architecture depends heavily on the situation where it is applied.

4.3.3 Buffering

Related to the problem of data copying is the problem of data buffering. At least one copy of the network layer packet (or a segment of it, depending on the mode of operation) needs to be buffered until all the needed processing has been performed. While operations are being carried out on part of the data (e.g., the destination address), data needs to be buffered until the results of the operation are available. It is most logical to buffer the data in the module where (virtual) reassembly of the network layer packet is performed, since all the data is available there anyway. The data also needs to be present in the module where the cell payload construction is performed, but this module is only known after part of the processing (routing) has already been performed (if it is not the same module).

If the CLS is operating in message mode, buffering of data for reassembly needs to be performed anyway. The reassembly buffer can also be used to store data until the results from the network layer processing are available. Also, the buffering for the traffic shaping function could be done in this buffer.

For a buffer operating in streaming mode, no reassembly buffer is needed. However, a buffer needs to be installed to store cells until processing has been completed. Different buffer strategies can be applied in this case. Data can be stored in a random access memory until the processing for the packet has been completed, and then be retrieved. Alternatively, cells can be stored in a FIFO (First In First Out) buffer for a fixed amount of time. After this time, the cell is retrieved from the buffer. Normally, the result of the packet processing is available at this time, and the cell is forwarded. In the exceptional case that the packet processing has not yet been completed, the cell, and all other cells from the packet are considered lost. If FIFO buffering is applied, an additional buffer is needed to store cells until the traffic shaping function schedules them for departure.

The buffering strategy to be applied depends on the used mode of operation. In case of streaming mode, several strategies are possible. An important advantage of FIFO buffering is that it does not require complicated buffer management schemes. However, it can on the other hand lead to additional delays and losses.

4.3.4 Integration with ATM Switch

Another implementation issue of interest is if, and to what extent, a CLS can be integrated with an ATM switch. Some of the functions to be performed in a CLS should also be performed in an ATM switch. These functions are those of the physical and ATM layer and switching and buffering. A CLS is integrated in an ATM switch if some of its functions are performed by modules that form also part of the ATM switch, i.e., that also handle regular ATM traffic.

What are the advantages of integrated implementation? Disregarding realization advantages (e.g., sharing power supplies), we can identify two types of advantages:

- performance
  The aim of providing a connectionless service on top of an ATM network is that the ATM network functions as a cell transport network, and that the CLSs perform routing and switching of the packets transported in the cells. Therefore, a CLS will be connected to a single ATM switch, or, to guarantee a higher availability, to a small number of them. This implies that traffic leaves the CLS through the same ATM switch as through which it arrived. Integrated implementation avoids that ATM cells are processed and switched more often than is needed. This results in improved performance in terms of throughput and delay.

- smooth evolution
  Connectionless services can be assumed to form one of the major applications in early ATM networks. As time proceeds, the demand for connectionless services may be gradually overtaken by a demand for connection-oriented services. Integrated ATM switches / CLSs can evolve by replacing only those modules that are specific to connectionless traffic, i.e., the modules performing AAL and network layer functions.

In defining the modules of a CLS in Section 4.1, we have already taken into account that some of the functions are performed in an ATM switch as well, and that they can hence be implemented in identical modules. These modules where the Input and Output Modules, responsible for the physical and ATM layer

processing, and the IS, responsible for switching and buffering of data. If we want to share the same modules by regular ATM traffic and connectionless traffic, one of the modules has to be able to identify connectionless traffic, and forward it to a module dedicated to this traffic. The IS is the most natural module for this, since its function is to switch, i.e., forward depending on some information, cells. Therefore, the use of the same IS is essential for an integrated ATM switch / CLS. In an ATM switch, this IS is called an ATM Switching Fabric ([140]).

Besides the IS, also the Input and Output Modules can be shared between connection-oriented and connectionless traffic. As far as the physical and ATM layer processing is concerned, there is no difference between the processing of both types of traffic. This more comprehensive integration is only possible if the Input and Output Modules are directly connected to the IS in the implementation architecture.

From the implementation architectures we have defined, all but the first one (which has no IS) allow for sharing the IS with an ATM switch. Architectures 3, 4, 5, and 6 can also share their Input and Output Modules with an ATM switch. Figure 4-20 shows the implementation architecture of a CLS (according to Architecture 6) with an ATM switch.

Whether or not a CLS will indeed be integrated with an ATM switch depends mainly on considerations beyond the scope of this dissertation. If the connectionless service is for instance operated by another organization than the underlying ATM network, there is a need for a clear interface between a CLS and an ATM switch. There will be a need for both integrated and separate CLSs.

4.4 Design Criteria applied to Implementation

In Chapter 1, we have identified a number of design criteria. The extent to which some of these criteria are met differs for different implementation architectures. In this section different architectures are compared with respect to a number of criteria. The important design criterion of performance is discussed in separate chapters (Chapters 5 and 6). Here we discuss availability in Section 4.4.1 and scalability in Section 4.4.2.

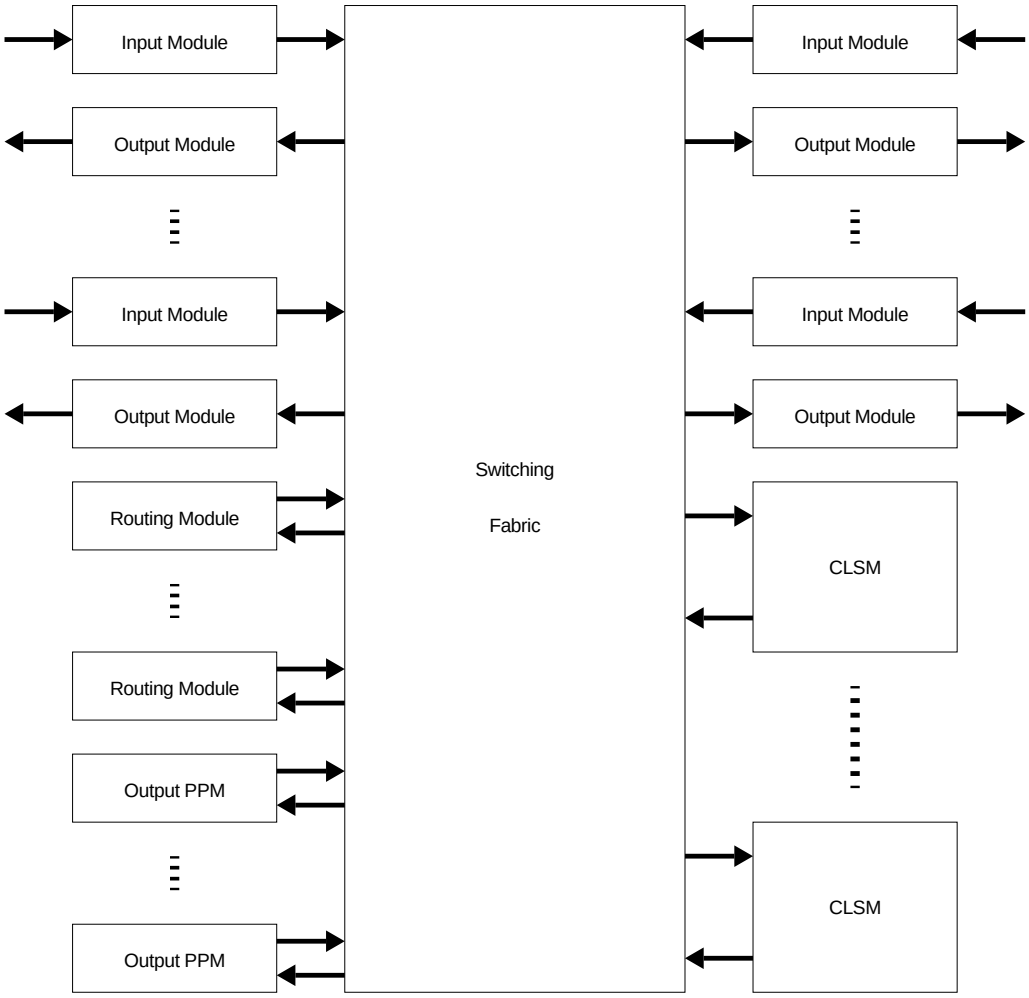


Figure 4-20: CLS integrated with an ATM Switch

4.4.1 Availability

Significant differences can be found in the ability of the different implementation architectures to cope with failing components. This ability is an important design criterion in the choice for a specific architecture. Here, the intent is to discuss qualitatively the inherent pros and cons of the architectures as far as their ability to cope with failure of the constituting modules is concerned. We do not have the intention to cover availability issues completely. Furthermore, we do not discuss the availability of the individual modules themselves.

We assume that modules either perform adequately or fail completely. A module exhibiting erroneous behaviour can be shut down by management if the erro-

neous behaviour is detected. We assume that appropriate failure detection mechanisms are present in the CLS.

The availability of a CLS can be enhanced by applying dynamic redundancy techniques ([47]). The general idea of these techniques is to reconfigure the system after failure of one of its components has been detected. For a CLS consisting of modules that may fail, this would imply that other modules can take over the processing if a module fails. Furthermore, the status information that was maintained in the failing module should be transferred to the module(s) that take over. Thus, the CLS as a whole can stay operational, although its performance may be degraded.

The ideal situation is where all modules are configured around a central IS, so that traffic can easily be rerouted to other modules by the IS, in case of module failure. Note that this scenario assumes a sufficient availability of the IS.

A module failing completely will not be able to transfer its status information to other modules any more. Therefore, one or more copies of this information should be maintained in other places. Three alternatives can be identified for maintaining copies:

1. one copy could be maintained centrally, by the management entities,
2. copies could be maintained by all modules with the same functionality as the concerned module, or
3. a copy could be maintained by a dedicated module, which is designated to take over if the concerned module fails.

Using the first alternative implies that the status information is downloaded to the module that takes over in case of failure. This reveals the disadvantage of the alternative; an additional delay is introduced between failure detection and recovery. The second alternative does not suffer from this disadvantage, but has to maintain a lot of status information in a lot of modules. This disadvantage is removed in the third alternative, although this alternative is somewhat more vulnerable to simultaneous failure of several modules.

Maintaining copies of all status information would require much interaction between the concerned module and the module where a copy is maintained. Since it is very likely that a number of cells get lost in case of module failure, it does not make sense to avoid loss of the other cells of the packets currently being processed. Therefore, only copies of static information, i.e., information that is not

updated after each packet (see Table 4-2), should be maintained. Upon start of service of the redundant module, the dynamic status information can be reset to initial values.

In case of ATM link failure, the ATM network is assumed to perform a reconfiguration. This could imply that incoming and outgoing ATM connections access the CLS through different Input and Output Modules. In this case, it is advantageous, if these connections can still be handled by the same CLSM. For that reason it is advantageous not to have a fixed CLSM per Input or Output Module.

Considering the implementation architectures identified in Section 4.1, we can see that Architecture 1 does not allow for any reconfiguration, since it does not have any redundant modules. The other architectures do allow for reconfiguration, if redundant modules are installed. Architecture 2 is not very flexible in its reconfiguration, since each Input or Output Module is connected to a single CLSM, and not via the IS to all. Therefore, the architectures 3 through 6 can be considered most suitable for increasing availability if individual modules are not sufficiently reliable.

4.4.2 Scalability

It is important to know to what extent a CLS can be dimensioned to cope with the expected traffic demands. Since these demands will change in time, it must be possible to extend an already installed CLS. Note that we concentrate on the scalability of a single CLS here. It should, of course, also be possible to extend the overall overlay network by installing new CLSs.

The throughput of a CLS can be increased by installing additional modules for processing. However, since only a single IS is present in (most of) the implementation architectures, the IS could form a bottleneck for the scalability of the CLS. It is important to use the IS efficiently if a CLS is to be extended. On the other hand, some ISs will to a certain extent be scalable themselves. It can be expected that the number of modules that can be attached to an IS can be increased. This possibility depends on the type of IS that is used ([140]). All implementation architectures but the first one allow for extension by installing extra modules, as long as the IS capacity is sufficient.

Another aspect of scalability is the balance between the load on different types of modules. Different types of modules perform different functions. The load on these modules depends on various parameters. For an Input or Output Module, the load depends on the number of ATM cells that should be processed per

second. For a packet processing module (the Output PPM), the load depends on the number of packets that should be processed per second. The same applies to the RM, of which the load also depends on the size of the network, because of the size of the routing tables.

In a CLS, the ratio between the number of cells offered per second and the number of packets offered per second is unpredictable, and will vary during operation. It depends on the fraction of the overall traffic in a switch that is connectionless traffic, and on the number of cells per packet. Furthermore, the network size will probably not grow proportionally with the traffic load. Therefore, it is very important for a CLS that parts that perform different types of functions can be scaled independently.

In the above perspective, Architecture 6 is most flexible in terms of scalability. Architectures 3, 4, and 5, which do not have a separate module for routing, are less flexible. Furthermore, Architecture 4 has the disadvantage that all cells are switched by the IS three times, which puts a heavy claim on the IS throughput. Architecture 2 only switches all cells once, but has the disadvantage that the number of CLSMs (including routing functions) can not be dimensioned independently from the number of Input and Output Modules. Architecture 1 is not scalable at all.

4.5 Summary and Concluding Remarks

In this chapter, we have discussed the problem of implementing a CLS. We have come up with a number of different implementation architectures. These range from a very basic one, consisting only of a single Input Module, a single Output Module, and a single CLSM, to a large scale one, aiming at high performance, and consisting of a number of Input and Output Modules, CLSMs, RMs, and Output PPMs, interconnected by a central IS.

A number of specific issues related to implementation have been discussed. We can conclude that it is still an open issue whether the message mode or the streaming mode of operation is preferable.

Other implementation issues, i.e., data copying and integration with an ATM switch, are strongly related to the used implementation architecture. Table 4-3 summarizes these relations on a scale from ‘bad’, via ‘poor’ and ‘reasonable’, to

	avoiding data copying	integration with ATM switch	availability	scalability
Architecture 1	good	not possible	bad	bad
Architecture 2	reasonable	poor	poor	poor
Architecture 3	reasonable	good	good	reasonable
Architecture 4	poor	good	good	reasonable
Architecture 5	reasonable	good	good	reasonable
Architecture 6	reasonable	good	good	good

Table 4-3: Summary of Comparison of Implementation Architectures

‘good’. The table also summarizes the suitability of the architectures according to availability and scalability criteria. Performance aspects of the architectures are not included in the table. Those will be discussed in the next chapters.

So here are the questions:
Is time long or is it wide?

Laurie Anderson - Bright Red

Chapter 5
Performance of a Connectionless Server

In previous chapters we have presented a number of design options for the provision of a connectionless service using ATM. These options have potentially a large impact on the performance of a CLS. In this chapter we quantify the effects of a choice for certain options as far as the performance is concerned.

Besides providing performance arguments for the choice of certain design options, this chapter will provide performance figures supporting the dimensioning of a CLS. It is important to know how certain system components must be dimensioned in order to support the required performance.

In Section 5.1, we will first pose some important questions that are of interest when designing or dimensioning a CLS. In order to be able to answer these questions, we have developed a queueing model of a CLS. This model, and its analysis, can be seen as a tool that can be used to predict the performance of a CLS. The model is presented in Section 5.2, and analysed in Section 5.3. In Section 5.4, we evaluate different implementation architectures and different modes of operation, using the tool. Furthermore, in order to validate our analysis with respect to accuracy of the results, we compare results, obtained with the tool, with those obtained from simulations. We end this chapter with a summary and concluding remarks, in Section 5.5.

5.1 Preliminaries

The purpose of this section is to identify issues that arise in the design of a CLS, and that have potentially a large impact on the performance of the CLS. First of all, the general notion *performance* must be clarified. What is the meaning of performance of a CLS? What are the measures we are interested in?

Performance analysis plays an important role in several phases of the design. Two different roles of performance analysis can be identified. The first one is to assess the quantitative aspects of different design choices. The second one is to deter-

mine the required capacity of certain resources. The first type of analysis, often has to cope with questions such as: Given a certain load to the system, and a certain amount of resources, what will the performance of the system be for design A and design B? Questions that have to be coped with by the second type of analysis are often of the form: What capacity X must a certain resource have in order to result in the required system performance, given a certain load to the system?

This chapter deals with both the selection of alternatives for the CLS design, and the CLS dimensioning. Therefore, the questions that will be posed in this section will be of both forms.

In the sequel of this section, we will first discuss performance measures that are of interest in the analysis of a CLS, in Section 5.1.1. Then, in Section 5.1.2, we will raise specific questions that need to be answered in this chapter. We raise questions related to the mode of operation of the CLS and to the used implementation architecture.

5.1.1 Performance Measures

Performance measures that will be considered can be classified into three groups:

- delay measures,
- loss measures, and
- throughput measures.

Delay measures express the time aspects of the operations that are performed on packets or cells. The end user will be interested in the time elapsing between the start of transmission of a packet at the sender and the complete receipt of the packet by the receiver. For a single CLS we are interested in the delay experienced by a cell between arrival and departure, i.e., the sojourn time. Furthermore, the time between departure of consecutive cells of a packet, the interdeparture time, is an interesting measure. In general, these delay measures can be described by a probability density distribution. Sometimes it suffices to give for instance the expectation of this distribution.

Loss is a phenomenon that occurs due to corruption of information during transmission or processing, or due to buffer overflow. Loss can be expressed by probabilities, or by more sophisticated measures, such as processes that capture also correlations between losses. The end users will be interested in the probability of

packet loss rather than cell loss, since in general the whole packet has to be retransmitted if a part of it is lost. Therefore, we consider the packet loss probability as an interesting performance measure for a CLS. Packet loss in a CLS will be discussed in Chapter 6.

Throughput can be defined as the amount of information per unit of time that can be handled by a CLS. Since some of the functions in a CLS operate on a per-packet basis, and some operate on a per-cell basis, both packets per second and cells per second are interesting units for throughput measures.

5.1.2 Research Questions

In Chapter 4, two modes of operation have been defined for a CLS. In message mode, a CLS completely reassembles an incoming packet before forwarding it to an outgoing connection. In streaming mode, cells of a packet are forwarded to the outgoing connection on an individual basis. On the basis of the first cell, the routing decision is taken, and this cell is forwarded to the relevant outgoing connection. Consecutive cells of the packet are forwarded to the same outgoing connection right away.

Q1: How do the sojourn times of cells compare for CLSs operating in streaming mode and in message mode?

It can be expected that a message mode CLS imposes a larger delay on a cell than a streaming mode CLS, since it involves an additional delay to reassemble the packet. On the other hand, we assume that a streaming mode CLS needs a fixed-size FIFO buffer that will impose a sufficient delay to have a very high probability that the routing information is available in time. This involves an additional delay compared to message mode, because cells have to wait for the entire delay of the FIFO buffer, regardless of when the routing information becomes available. We are interested in the sojourn times of cells during a high load situation.

For the end-to-end delay of a packet it is also interesting to know more about the time between departures from the CLS of consecutive cells of the same packet. It can be expected that the interdeparture time of cells within a packet is smaller for a message mode CLS than for a streaming mode CLS. This brings us to the next question.

Q2: How do the interdeparture times of cells within a packet compare for CLSs operating in streaming mode and in message mode?

A third performance measure of importance is the maximum throughput that can be obtained for a given amount of resources.

Q3: How do the throughputs compare for CLSs operating in streaming mode and in message mode?

An important dimensioning problem in the design of a message mode CLS is the dimensioning of the buffer that is used to reassemble the incoming packets. We assume that these buffers will be dimensioned in such a way that the packet loss probability meets certain requirements.

Q4: How large should the reassembly buffer in a message mode CLS be in order to meet the packet loss requirements?

The dimensioning of other buffers in the CLS, i.e., the streaming mode buffer and the traffic shaping buffer has already been dealt with elsewhere, e.g., in [28] and [124] respectively.

Associated with an outgoing ATM connection to another node, is a traffic descriptor, as part of a traffic contract, which characterizes the traffic that can be expected on the connection. This descriptor is used by the ATM network to reserve the proper resources for a connection. The parameters of this traffic descriptor are also used by the traffic shaping function, to shape the outgoing traffic on a connection, so that the traffic contract is not violated. Cells that would violate the contract are delayed until they can be safely transmitted on the connection. The traffic parameters should be such that the traffic for the connection can be accommodated without imposing too high delays. This leads to the following question:

Q5: What values should be used for the parameters of the traffic descriptor, agreed on for an outgoing ATM connection?

Clearly, these parameters should not only be optimized for low traffic shaping delay. They should also be such that excessive reservation of bandwidth in the ATM network is avoided.

In Chapter 4 we have introduced a number of different architectures for the implementation of a CLS. Many of the consequences of the choice for a particular architecture have been discussed in that chapter. The choice for one of the architectures also has an impact on the performance of the CLS. In particular, the influence of the choice on delay and achievable throughput needs to investigated.

Q6: What is the maximum achievable throughput of the different implementation architectures?

Q7: How do the sojourn times of cells for the different implementation architectures compare?

The research questions posed in this section can not be answered independently, e.g., the sojourn times of cells depend on the used traffic parameters. We will address them in coherence with each other. The questions will heavily influence the modelling and analysis in this and the following chapter, i.e., the models will be specifically designed for addressing the above questions.

5.2 Modelling

In order to obtain insight in the delay experienced by cells in a CLS (Question 1 and 7), and also in other delay and throughput related issues (Question 2, 3, 5, and 6), we have developed a queueing model of a CLS. Evaluation of a CLS using the model can be done by simulation or analytically, using approximate queueing network analysis techniques. The approximate analysis is presented in Section 5.3. Section 5.4 presents the evaluation of the model using both the approximate analysis and simulations.

In Section 5.2.1 we will first introduce some queueing model components and notation that we will use throughout the rest of the chapter. In Section 5.2.2, we first classify the implementation architectures, identified in the previous chapter, and introduce a parameter that captures the difference between the classes. Furthermore, we present a global model of a CLS, which consists of four submodels. Section 5.2.3 describes the modelling of the assumed traffic load to the CLS. Sections 5.2.4 to 5.2.7 work out the submodels that have been identified in the global model. For some of the submodels, it is necessary to distinguish between CLSs operating in streaming and in message mode. Finally, in Section 5.2.8, the overall model is presented.

5.2.1 Notation

The use of queueing models is very widespread ([89]). Different notations are used to display these models. Since we introduce some extensions to standard queueing networks, it is wise to start with introducing the notation used throughout this chapter.

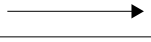
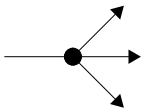
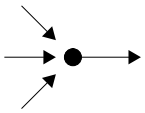
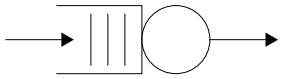
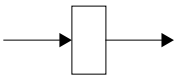
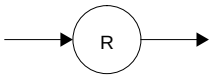
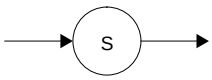
	traffic stream
	splitting traffic stream
	merging traffic streams
	service node
	delay node
	reassembly node
	segmentation node

Table 5-1: Queueing Model Notation

In a queueing model, we can identify nodes and customers. Nodes represent parts of the real system. Customers represent the entities the system operates on. In the CLS model, they represent cells or complete packets. Customers proceed between nodes, which perform certain actions on them. These actions will be discussed later on.

A number of symbols will be used to represent the components of queueing models (Table 5-1). A *traffic stream* represents a series of consecutive customers, proceeding from one node to another. It is characterized by the times elapsing between consecutive customer passing by. It is assumed that it does not take time for a customer to proceed from one node to another.

Two operations on traffic streams are considered. Customers do not experience any delay during these operations, only decisions regarding the node to proceed to are taken. In a *splitting traffic stream*, each incoming customer proceeds to only one of the outgoing traffic streams. The outgoing traffic stream is chosen randomly, and independently for each customer. In case of *merging traffic streams*, customers from all incoming streams proceed to the same outgoing traffic stream.

n	number of (Input and Output) CLSMs per CLS
k	number of adjacent nodes in the connectionless overlay network
m	number of connections per CLSM
p_{arrival}	probability of a cell arrival at a CLSM in a time-slot
ρ_c	utilization of the bandwidth of a connection
ν	peak cell rate of a connection
ν_{total}	sum of peak cell rates of all connections to / from the same adjacent node
μ_{cip}	reciprocal of the cell input processing time
L	random variable describing the packet length
l	mean packet length
c_L^2	squared coefficient of variation of the packet length
b	size of the FIFO buffer in streaming mode
μ_{cop}	cell output processing rate

Table 5-2: CLS Model Parameters

In a *service node*, a customer receives service, which will take a certain amount of time. This service time can be seen as the time needed to process a cell, packet, or piece of control information. If no server is available upon arrival because all servers are busy serving other customers, a customer may have to wait in a queue before his service starts. A service node is characterized by the duration of the service for a customer, which is generally a random variable. Furthermore, it is characterized by the number of servers that provide service and the order in which arriving customers are served. In our model, all service nodes have a single server, and we assume a 'first-come, first-served' service discipline.

A *delay node* imposes a certain delay on customers. The reason for this delay can be that customers are buffered for some time, or that customers receive service and there is always a server available. Indeed, a delay node behaves the same as a service node with an infinite number of servers.

As stated before, a customer can represent a cell or a packet, depending on the part of the system we consider. In order to transform a number of cells into a single packet, reassembly is performed in a CLS. A *reassembly node*[1] in a queueing model transforms a number of consecutive customers from the incoming traffic stream into a single customer on the outgoing traffic stream. Thus it can trans-

[1] Reassembly is also called customer combination in publications on queueing networks ([54], [143]). Segmentation is also called customer creation.

form customers representing cells to customers representing packets. The transformation from cell to packet takes time. An incoming customer has to wait until all other customers representing cells of the same packet have arrived, after which the customer representing the packet can depart.

In order to model the reverse process of reassembly, we also use a *segmentation node*. Upon arrival of a customer in this node, a number of customers departs simultaneously on the outgoing traffic stream. Both the reassembly and the segmentation node are characterized by a random variable. For the reassembly node, this is the number of customers to combine in a departure. For the segmentation node, this is the number of cells to generate upon arrival of a customer. In both cases, the random variable models the packet length.

The parameters needed to characterize the models in this chapter are shown in Table 5-2. These parameters will be explained in the next sections.

5.2.2 Global Model

The purpose of the model presented in this chapter is to obtain insight in the delay experienced by cells in a CLS. Therefore, we focus on those aspects of the system that are mainly responsible for this delay. Furthermore, the main interest is to compare different alternatives under different load situations.

The main alternatives that will be compared are the following:

- message mode vs. streaming mode of operation
  In message mode, cells experience delay because they must wait until the packet they belong to has been completely reassembled. In streaming mode a fixed size FIFO buffer is used to store cells until routing information is available, which also induces delay.

- different implementation architectures
  The implementation architecture has an impact on the number of outgoing connections a CLS should have, as will be shown later. Two classes of architectures can be identified in this respect. The number of outgoing connections influences the delay.

In the model, we only consider the part of the CLS from the (Input) CLSM, where cells first enter until the (Output) CLSM, where the cells depart, i.e., we only consider the part of the CLS that performs the AAL and network layer processing. The Input and Output Modules are not modelled here, since their behaviour does not depend on the alternatives we want to compare. They are

expected to impose a small fixed delay on the cells for processing of the ATM header. Furthermore, the behaviour of the IS is not incorporated in the models. The IS is not essential to the behaviour of a CLS in different modes of operation, or with different implementation architectures. We assume that the IS is dimensioned in such a way that it has sufficient capacity, even if cells are switched over the IS several times, e.g., in Implementation Architecture 4. Furthermore, it is expected that cells do not experience significant delays during this switching. In fact, a lot of work on the performance of ATM based Interconnection Structures has been done already. See for instance [106] for an overview.

In modelling the (Input and Output) CLSM, we can identify a number of processes that have to be performed successively (see Figure 5-1). Upon arrival of a cell in the (Input) CLSM, *cell input processing* is performed. This process performs the processing related to the AAL, e.g. error detection. Furthermore, it extracts the network layer PCI from the arriving cells, and forwards it to the packet processing. After that, the cells are forwarded to the buffering process. *Packet processing* is concerned with address screening, address validation, access class enforcement, and routing. It needs the network layer PCI, and results in an indication of the outgoing ATM connection, on which the cells of the packet must be transferred.

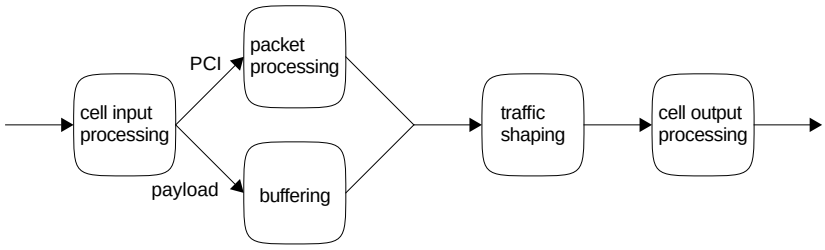


Figure 5-1: Processes in a CLSM

During the packet processing, the cells themselves are stored by the *buffering* process. The type of buffering depends on the used mode of operation. In message mode, a *reassembly buffer* is used, in which cells are stored until all the cells of the packet they belong to have arrived. Thereafter, these cells are forwarded to the traffic shaping process, provided that the result from the packet processing, i.e., the relevant outgoing ATM connection, is available. If this result is not available, the cells wait in the reassembly buffer until it becomes available. In streaming mode, a *FIFO buffer* is used. In this buffer, cells are stored for a fixed time. After this time, they are forwarded to the traffic shaping process, provided

that the result from packet processing is available. If this result is not available, the cell, and all consecutive cells of the same packet are considered lost.

Traffic shaping is performed for each outgoing ATM connection separately. It ensures that the stream of cells, transferred on an outgoing ATM connection, conforms to the traffic descriptor that is agreed on for the connection. As stated in Section 3.2.4, only a single traffic parameter, the peak cell rate, has currently been defined by the ITU. The traffic shaper delays cells in such a way that the peak cell rate for an outgoing connection is not exceeded. After that, the cells are forwarded to the cell output processing.

Cell output processing constructs the proper AAL PCI, and forwards the cells onto the IS. Since it may receive cells from different traffic shapers, i.e., for different outgoing ATM connections, simultaneously, this process will also arbitrate the access to the IS. Cells may have to be queued before they can be forwarded to the IS, because other cells may have to be forwarded first.

The implementation architectures that have been identified in Chapter 4 can be divided into two classes. These two different classes result in differences in the modelling of a CLS. Different implementation architectures within a class are modelled by the same approximate model. Simulation experiments have shown that there are no significant differences in the delay between architectures within a class. Let us now explain the difference between the two classes, and show how this difference is reflected in the modelling.

Class A

In implementation architectures 1, 3, 5, and 6, the AAL and network layer processing is performed in a single CLSM, for a given packet[2]. This packet can be destined to any node that should be reachable as a next hop in the connectionless overlay network (either a CLS or an end-system). Therefore, ATM connections from the concerned CLSM to all possible adjacent nodes must be maintained. Packets arriving in another CLSM that are destined for the same adjacent node must use another ATM connection. As a consequence, a CLS, consisting of n CLSMs, must maintain (at least) n ATM connections to the same adjacent node.

The reason that each CLSM needs to have its own connection to a certain adjacent node is that connections cannot be merged after traffic shaping has taken place.

[2] In fact, part of the network processing can also be performed in another module, i.e., an Output PPM or RM. However, as we will see, this is not relevant for the modelling in this chapter

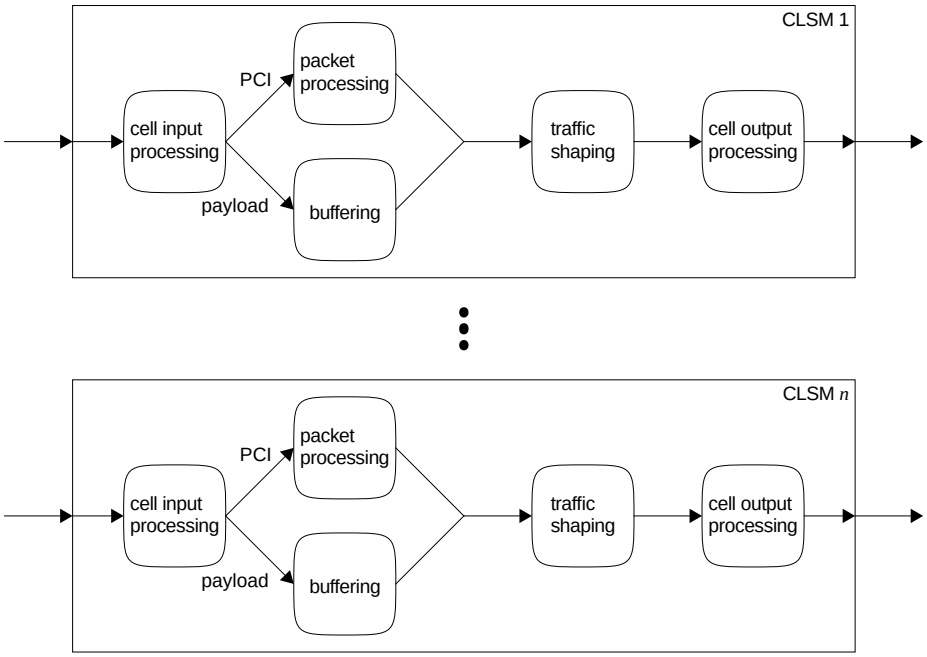


Figure 5-2: Class A: Multiple Connections to the next Node

Traffic shaping for an outgoing connection must be implemented in a single place in the CLS, because a traffic shaper must update state information for each cell departure. Distributed implementation of a traffic shaper would lead to an unacceptable overhead for the exchange of state information, and furthermore the immediate availability of the information could not be guaranteed. As a result, traffic streams that should be transferred on the same connection should be merged before the traffic shaping is performed[3].

This class of implementation architectures with multiple connections to an adjacent node, will be referred to as *Class A*. Figure 5-2 illustrates where the different processes are performed in a Class A architecture. Only traffic destined for the same adjacent node and its processing is shown in the figure. As shown, packets for this node may arrive in any of the *n* CLSMs. All five processes that have been identified above, are performed in the CLSM where a packet arrives. Since traffic shaping is also performed in that CLSM, the CLSM has its own connection to the adjacent node.

[3] In principle, it is possible to use a distributed implementation of the traffic shaping function, where the bandwidth of the outgoing connection is shared between the traffic shapers on a more permanent basis. But there is no basic difference between this method and having a single connection per traffic shaper, other than the use of connection identifiers.

Class B

In implementation architectures 2 and 4, the AAL and network layer processing is distributed over an Input and an Output CLSM. After cell input processing, buffering, and packet processing has been performed in an Input CLSM, the required outgoing ATM connection is known. Based on this knowledge, the cells of a packet are forwarded to a specific Output CLSM, in which the traffic shaping for that connection is performed. Cell output processing is also performed in this Output CLSM. Consequently, it suffices to have a single ATM connection, for which traffic shaping is performed in one of the Output CLSMs, from a CLS to the next node in the overlay network. The concerned Output CLSM is dedicated to a number of adjacent nodes to which packets must be sent. Traffic arriving at an Input CLSM can be forwarded to the proper Output CLSM, depending on the required destination.

This class of implementation architectures with a single connection to an adjacent node, will be referred to as *Class B*. This class is illustrated in Figure 5-3. Again, only traffic destined for the same adjacent node and its processing is shown in the figure. Packets, destined for this node, may arrive in all Input CLSMs. After processing in the Input CLSM has been completed, they are forwarded to the Output CLSM connected to the adjacent node (Output CLSM 1 in the figure). Here, all traffic, coming from different Input CLSMs and destined for the same node, is merged, and traffic shaping is performed on the merged stream.

Let k denote the number of adjacent nodes, which should be reached from the CLS. We have already introduced n as the number of CLSMs in Model A. It denotes also the number of Input CLSMs and the number of Output CLSMs in Model B. We can state for m_A, the number of outgoing ATM connections that should originate from a single CLSM in a Class A architecture:

$$m_A = k \ . \tag{5.1}$$

For a Class B architecture, the connections to the adjacent nodes can be equally divided over the Output CLSMs, i.e., we can state for m_B, the number outgoing connections that should originate from a single Output CLSM:

$$m_B = k/n \ , \tag{5.2}$$

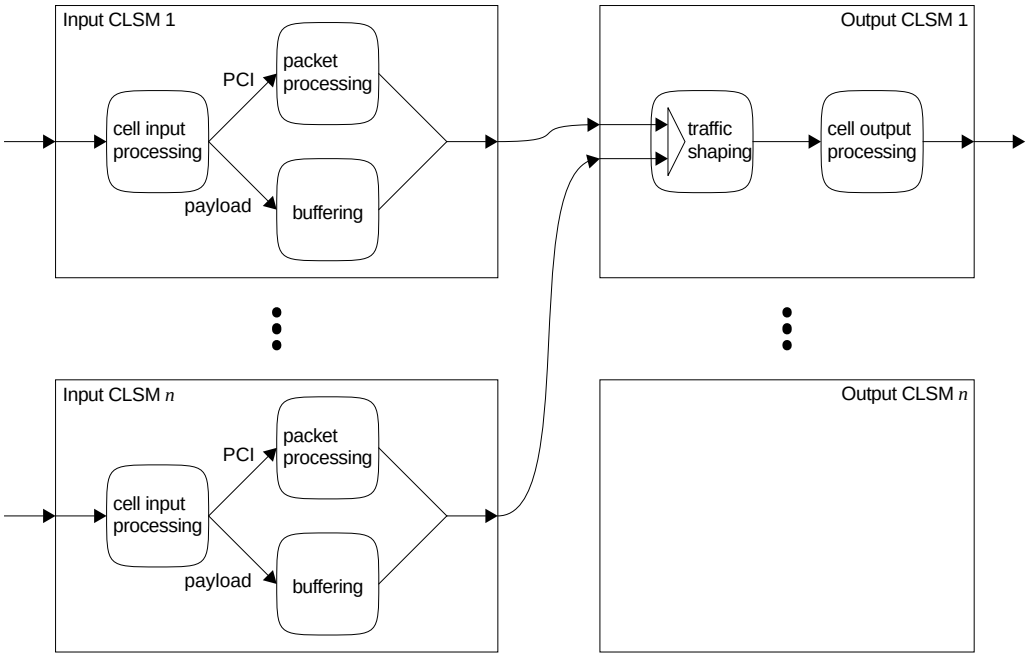


Figure 5-3: Class B: Single Connection to the next Node

provided that k is an integer multiple of n. We will omit the subscript of m if we do not address any of the two classes in particular. In the model, the difference between Class A and Class B architectures can be totally captured by the value of m.

The total number of connections originated in a CLS is $n \times m$. We assume that the overlay network is homogeneous, i.e., each CLS has an equal number of adjacent nodes, so that the total number of connections terminated in a CLS, i.e., entering a CLS, equals $n \times m$ as well. Assuming, that they are equally divided over the (Input) CLSMs, the number of incoming connections per (Input) CLSM equals m again.

In Figure 5-1, we have considered the successive processes that must be performed for the cells of a packets in a CLS. From these processes, we derive a global model (Figure 5-4). We identify a number of submodels, each representing a part of a CLSM that is responsible for performing one of the processes for all cells or packets that visit the CLSM.

Four submodels are identified in this global model, corresponding to four of the processes. No submodel for packet processing has been identified in this model, because simulation experiments have shown that the delay experienced by a cell is almost completely determined by the delay caused by the buffering, and not by

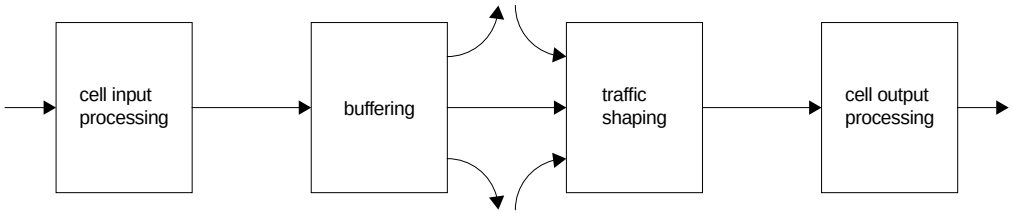


Figure 5-4: Global Model of a CLS

the delay caused by the packet processing, which is performed simultaneously. Therefore, the model presented here, and analysed in Section 5.3, only models buffering. The simulation model used in Section 5.4 does model packet processing as well.

Between the buffering and traffic shaping submodels, traffic may leave or enter the model. This represents the traffic that goes to or comes from other CLSMs, in case of a Class B architecture. In case of a Class A architecture, these traffic streams do not exist.

Let us now discuss the various submodels in detail. Before doing so, we consider the stream of cells entering the cell input processing submodel.

5.2.3 Arrivals

The unit of time we adopt in this model, is the time needed to transmit one ATM cell on a link (a cell time). On a 155.520 Mbits/s ATM link, a cell time equals approximately 2.7μs . The interface between a CLSM and the IS is assumed to work at the same speed.

It is assumed in our model that cells arrive to the cell input processing submodel according to a Bernoulli process. In a time-slot, a cell arrives in the CLSM with probability p_{arrival}. As a result, the interarrival time between two consecutively arriving cells is geometrically distributed with a mean of $1/p_{\text{arrival}}$ cell times. An arriving cell is assumed to belong to each of the incoming connections with probability $1/m$.

We define the following relation for the arrival probability:

$$p_{\text{arrival}} = m\rho_c \nu \ , \tag{5.3}$$

where ν denotes the peak cell rate, that is agreed on in the traffic contract for a single connection, and ρ_c denotes the utilization of this connection. Note that, in general, ν corresponds also to the fraction of the capacity (bandwidth) of an ATM

link that is assigned to a connection, since the sum of the peak cell rates of all connections on a link should be at most one, in which case all time-slots (one per cell time) are reserved for these connections.

The use of this simple arrival model is inspired by the fact that we are interested in the delay of a CLS during periods of high loads. We consider the steady state behaviour of the system at a relatively short time scale. Long term correlations between arrivals, such as the ones observed in ([94]) do not play a role at this time scale.

5.2.4 Cell Input Processing

The cell input processing submodel, models the processing of incoming cells for the SAR sublayer. Typically, this processing is performed on the fly, and induces only a small fixed delay. Therefore, the model is very simple (Figure 5-5). It consists of a single delay node, with a constant delay of $1/\mu_{cip}$ time-slots. All arriving customers, which represent the consecutive cells to be processed, experience this delay.

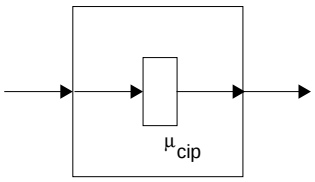


Figure 5-5: Cell Input Processing Submodel

5.2.5 Buffering

The buffering submodel models the delay experienced by cells in either the reassembly buffer or the FIFO buffer, depending on the mode of operation. The model consists of a number of reassembly nodes, one for each incoming connection (Figure 5-6). As stated in Section 5.2.3, an arriving cell belongs to a certain connection with probability $1/m$. Therefore, the incoming traffic stream is split over the m reassembly nodes randomly, according to equal probabilities $1/m$.

The customers leaving the reassembly node represent packets. Although we do not model the processing time for routing, we do model the routing decisions. It is assumed that a packet is routed randomly to one of the k destinations, i.e., we assume homogeneous traffic. This is modelled by the splitting of the traffic stream after the reassembly node. The traffic stream leaving from a reassembly node is split over k outgoing traffic streams according to equal probabilities $1/k$.

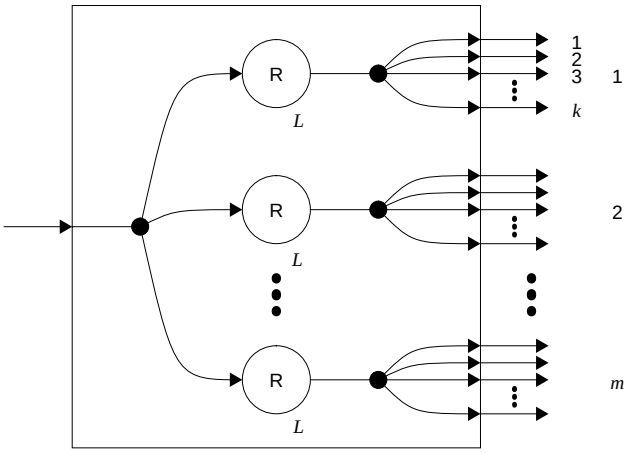


Figure 5-6: Buffering Submodel

Thus, the consecutive customers of a single traffic stream leaving the buffering submodel represent packets with the same incoming and outgoing connection. Note that all outgoing traffic streams proceed to the same traffic shaping submodel if a Class A architecture is modelled. If a Class B architecture is modelled, for each of the m incoming connections, only m of the k outgoing streams proceed to the same buffering submodel. The other streams proceed to other submodels, representing other CLSMs.

Let us now describe the exact operation of the reassembly node. It is different for the message mode of operation and the streaming mode of operation.

Message Mode. The message mode operation is as follows. A customer, representing a packet, departs only as soon as a certain number of customers, representing cells, have arrived (Figure 5-7). Arriving customers wait until the last customer for that particular departure has arrived. With the discrete random variable $L = 1, 2, \ldots$, we denote the number of arriving customers needed to form a departing customer.

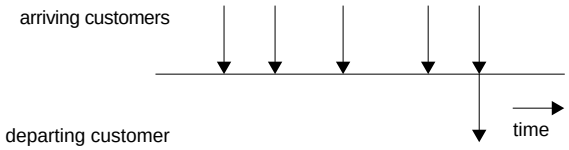


Figure 5-7: Behaviour of a Reassembly Node in Message Mode

Streaming Mode. The streaming mode operation is different. The reassembly node now also combines a sequence of arriving customers into a single departing customer, as in message mode. However, the departure time of the departing

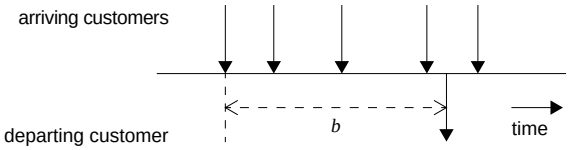


Figure 5-8: Behaviour of a Reassembly Node in Streaming Mode

customer does not equal the arrival time of the last customer of the sequence. The departing customer departs b units of time after the arrival of the first customer of the sequence (Figure 5-8). The meaning of L is the same as for the message mode.

Clearly, the message mode of a reassembly node models the situation where a packet is completely reassembled from the arriving cell before it is forwarded. The distribution of L is the packet length distribution in cells. We denote the mean of this distribution as l, and its squared coefficient of variation as c_L^2.

The streaming mode of a reassembly node models the situation where the cells of a packet are delayed for a fixed time, using a FIFO buffer of b cells. After this delay, the results from routing and other packet processing functions should be available, and the cell is forwarded. We have chosen to let only a single customer depart as the representation of all cells of the packet to facilitate the analysis in the next section. This customer models the departure of the first cell of a packet from the FIFO buffer. The arrival times of the other customers of the packet, relative to the arrival time of the first one, are preserved and used later on in the model, during the modelling of the (virtual) segmentation.

5.2.6 Traffic Shaping

The traffic shaping submodel models the delay experienced by users, caused by the traffic shaping function. This function is needed to avoid violation of the traffic contract on an outgoing connection. In Section 3.2.4 we have introduced the peak cell rate as the only traffic parameter defined up to now. In order to guarantee that the peak cell rate on a connection is not exceeded, the traffic shaper must ensure that the time between two successive departures is never less than the reciprocal of the peak cell rate.

The behaviour of the traffic shaper can be modelled by a special type of service node. This node has a zero service time for its customers, but is unavailable for a certain fixed service time after each service completion, i.e., after each departure. Consequently, customers experience only waiting time and no service time in this service node. The waiting time is the same as the waiting time experienced in a

single server queue with a deterministic service time. Also the successive interdeparture times are the same for the service node with zero service time and vacations, and the service node with deterministic service time. In fact, the only difference between the two is that in a regular service node, all customers depart exactly one service time later than in a node that represents a traffic shaper. Therefore, for our modelling, we use a regular service node and deduct the service time from the sojourn time of the node.

We have defined ν as the peak cell rate agreed on for an incoming connection. We assume that an outgoing connection has the same peak cell rate, ν. The traffic shaping is modelled by m service nodes with deterministic service time $1/\nu$, one for each outgoing connection.

For each of the m outgoing connections, k traffic streams enter the submodel, one for every adjacent node (Figure 5-9). If $m = k$, all traffic streams come from the same buffering submodel, and a Class A architecture is modelled. If m is smaller, they come from different submodels, representing different CLSMs, and a Class B architecture is modelled.

The incoming traffic streams for an outgoing connection are first merged to a single stream. Recall that the customers in this stream represent packets. Before going through the traffic shaper, these packets must be (virtually) segmented into cells again. This is modelled by a segmentation node. In the model, customers departing from a segmentation node are forwarded to the corresponding service node modelling a traffic shaper. Customers departing from the service node are proceeding to the cell output processing submodel.

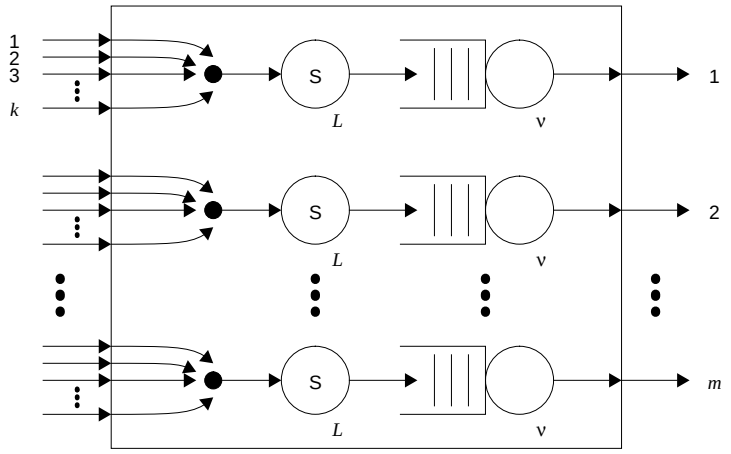


Figure 5-9: Traffic Shaping Submodel

Let us now describe the detailed behaviour of the segmentation node. This operation is different for the message mode of operation and the streaming mode of operation.

Message Mode. In message mode the operation of a segmentation node is as follows. Upon arrival of a customer in the node, a batch of customers departs instantaneously. In the simulation model (Section 5.4.1), the size of this batch equals the number of customers that were combined in the reassembly node for the departure of the same customer. In the model as it is analysed in the next section, the size of the batch is a random variable that is distributed according to the distribution of L.

Streaming Mode. The streaming mode behaviour of the segmentation node is different. It can best be explained in coherence with the behaviour of the reassembly node in streaming mode (Section 5.2.5). In the reassembly node, a single customer departs after the arrival of a sequence of customers. The departure is b cell times after the arrival of the first customer of the sequence. It was stated that the information of the arrival times of the other customers of the sequence, relative to the arrival time of the first one, is preserved. This information is used in a successive segmentation node, so that the original sequence of customers, with the original interdeparture times can be regenerated. As a result the combination of a reassembly node and a successive segmentation node delays all arriving customers exactly b cell times (see Figure 5-10).

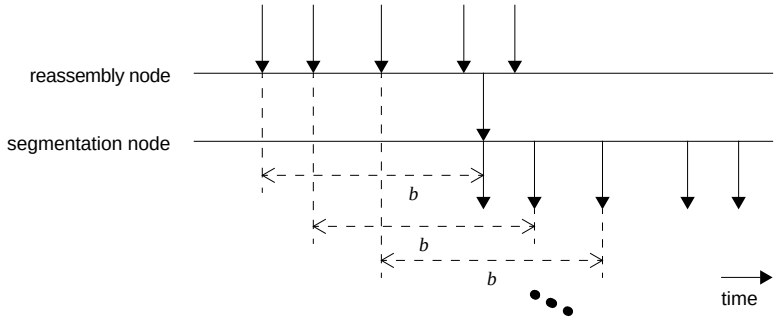


Figure 5-10: Behaviour of the Combination of a Reassembly Node and a Successive Segmentation Node in Streaming Mode

The reason to model such a simple behaviour with such a complicated construction is that in our model, traffic streams are split between the reassembly and the segmentation node. In the class of queueing networks we are able to analyse (Section 5.3.1), this splitting occurs according to random, independent routing

decisions for consecutive customers. By subsequently combining a sequence of customers into a single one, routing this customer, and regenerating the original sequence again, we accomplish that all customers in the sequence, representing the cells of a single packet, are routed to the same node.

Hence, the behaviour of a segmentation node in streaming mode is defined as follows. A number of customers departs upon the arrival of a single customer. However, only the first one of these customers departs instantaneously. The others depart one by one with nonzero interdeparture times. These successive interdeparture times equal the successive interarrival times as they have been observed at the corresponding reassembly node for the same sequence of customers.

In the approximate model, which is analysed in the next section, the number of customers to be regenerated does not actually equal the number that has been combined at the reassembly node. In fact, both are taken from the same distribution (the distribution of L). Similarly, the interdeparture times for the sequence of customers at the segmentation node do not actually equal the interarrival times at the reassembly node. They are taken from the interarrival time distribution that is observed at the reassembly node. In the simulation model (Section 5.4.1), the number of customers to be regenerated, and their interdeparture times, equal the number of users that are combined at the reassembly node, and their interarrival times.

5.2.7 Cell Output Processing

The final submodel constituting the overall model is the cell output processing submodel. It models the processing of outgoing cells for the SAR sublayer, and the multiplexing of cells from different connections on a single ATM link or on a single port of the IS. By definition, one cell can be transmitted on the ATM link per time-unit. We assume that the SAR processing is performed on the fly during this time-unit.

The resulting model is straightforward (Figure 5-11). Incoming cell streams, representing different outgoing connections, are merged. The customers on the merged traffic stream, which represent cells, queue for a service node. This service node is a single server queue with a deterministic service time, $1/\mu_{cop}$, of 1 cell time.

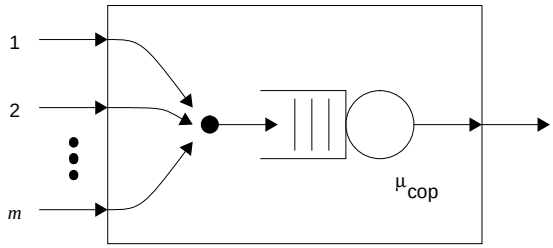


Figure 5-11: Cell Output Processing Submodel

5.2.8 Overall Model

In the previous subsections, we have defined the submodels in detail. This allows us to construct the overall queueing network that models the CLS. Figure 5-12 depicts the overall model as far as it is on the path of a single traffic stream. At some points in the model, traffic is split over several streams. These streams proceed through the same types of nodes as the considered traffic stream. Furthermore, streams, which have passed the same types of nodes, are merged with the considered traffic stream. The number of identical streams in which a stream is split, and the number of identical streams which are merged are indicated in the figure.

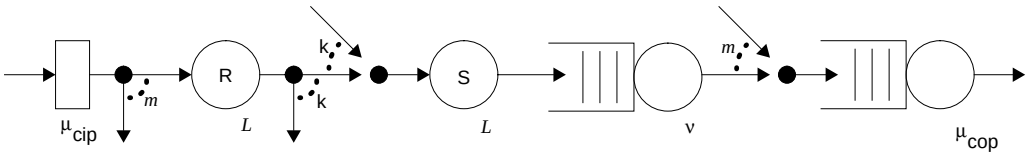


Figure 5-12: Overall Model for a Single Traffic Stream

5.3 Analysis

In the previous section, we presented a model of a CLS, which enables us to predict the delay performance under different circumstances and design alternatives. One way of obtaining performance figures from the model is by simulation. However, simulation is a costly process and it gives only estimates of the performance measures. Alternatively, we can obtain results from the model using analytical techniques. This has the advantage that we can obtain results quickly, and that we have explicit expressions for the performance measures. This provides insight in the behaviour of the model, and as a consequence in the behaviour of the system. However, some extra simplifying assumptions, and approximations have to be made.

$\lambda_{arrival}$, $c^2_{arrival}$	rate and squared coefficient of variation of the traffic stream entering the cell input processing submodel
λ_{cip} , c^2_{cip}	rate and squared coefficient of variation of the traffic stream departing from the cell input processing submodel
λ_{buf} , c^2_{buf}	rate and squared coefficient of variation of a traffic stream departing from the buffering submodel
λ_{s_in} , $c^2_{s\_in}$	rate and squared coefficient of variation of the traffic stream entering a segmentation node in the traffic shaping submodel
λ_{ts} , c^2_{ts}	rate and squared coefficient of variation of a traffic stream departing from the traffic shaping submodel
λ_{srv_in} , $c^2_{srv\_in}$	rate and squared coefficient of variation of the traffic stream entering the service node in the cell output processing submodel

Table 5-3: Characterization of Traffic Streams

In this section, the analysis of the queueing model of the CLS is presented. The analysis is based on work presented in [143], which is used in the software tool QNA (Queueing Network Analyzer). Furthermore, we have extended the analysis technique used in QNA. This extension has been described in [54] and [56].

For the purpose of our analysis, we have to characterize the traffic streams at the different places in the model. Table 5-3 summarizes the parameters that have been defined for this characterization. The parameters will be introduced later on in this section.

The structure of this section is as follows. First, in Section 5.3.1 we overview QNA, and give the results from QNA that will be used in our analysis. In Section 5.3.2 we briefly outline our extension to QNA, and present the results, which will be used for the analysis of the reassembly and segmentation nodes. In Sections 5.3.3 through 5.3.7 we derive the expressions for the arrival process and the various submodels. Finally, in Section 5.3.8 we present the expression for the mean sojourn times of a CLS.

5.3.1 QNA

QNA is the abbreviation of Queueing Network Analyzer, and is the name of a tool for the approximate analysis of queueing networks. It has been developed at AT&T Bell Laboratories by Whitt et al. ([143]). The expressions used to derive performance measures for queueing networks have been described in detail in

the same reference. In the analysis of the queueing network model of a CLS, we use some of these expressions. In this subsection we will give a brief overview of the QNA analysis technique, and present the expressions that will be used in our analysis.

The queueing network models considered in QNA are the same ones as those considered here. However, QNA does not allow for reassembly and segmentation nodes. The successive interarrival times between customers entering the network at a certain node are assumed to be random variables, which are mutually independent and identically distributed (i.i.d.). Successive service times at a node are also assumed to be i.i.d. random variables.

The results that can be obtained from the QNA analysis consist of the first two moments of the number of customers in the system, and the first two moments of the sojourn time of a customer. These results are obtained by first calculating these measures for each node individually. In other words, for each node the first two moments of the number of customers in the node, and the first two moments of the waiting time are calculated.

In order to be able to calculate these measures, the flow of customers through the queueing network has to be determined first. This results in a characterization of the interarrival process at each node. This interarrival process is approximated by a renewal process, which is characterized by its rate (λ) and its squared coefficient of variation (c^2).

Calculating performance measures from the specification of the queueing network involves a number of steps. Depending on the structure of the queueing network not all these steps are needed. We use the following steps in our analysis:

1. Calculation of traffic flow equations;

   The three basic operations on the traffic stream in a queueing network are: merging of streams, splitting of a stream, and the operation of a service node[4]. For each of these operations the arrival process is assumed to be a renewal process, and the departure process is approximated by a renewal process. The relationship between the arrival rate and the departure rate is given by linear equations. Thus, the traffic rates in the network can be determined by solving a

[4] QNA allows the specification of a customer creation / combination factor for a service node, which specifies the ratio between arriving and departing customers. However, since this is too limited in both its modelling capabilities and its analysis, we do not use it. Instead, we introduce two new types of nodes, segmentation nodes and reassembly nodes, to model the phenomena of customer creation and combination.

system of linear equations. Similarly, the traffic variability (squared coefficients of variation) can be determined by solving another system of linear equations.

2. Calculation of performance measures per node;

   Given the first two moments of the interarrival time distribution and the service time distribution in a node, several performance measures can be calculated. In particular the mean and variance of the waiting time, the sojourn time, and the number of customers in the node are derived.

3. Calculation of overall performance measures;

   Once the performance measures per node are determined, performance measures for the overall queueing network can be derived. The mean and variance of the number of customers in the network is determined. The mean and variance of the sojourn time of the aggregate customer is determined.

A number of expression are given for these steps in [143]. Here we only give the expressions as far as they are used in this chapter. For step 1 the following expressions are used. Let λ_{in} and c_{in}^2 be the rate and the squared coefficient of variation of the traffic stream before the operation. For the traffic rate after a service node (λ_{serve}), we have

$$\lambda_{serve} = \lambda_{in} , \tag{5.4}$$

i.e., the output rate of a service node equals its input rate. The utilization of the server (ρ) is defined as

$$\rho = \frac{\lambda_{in}}{\mu} , \tag{5.5}$$

where μ is the service rate of the server. The squared coefficient of variation of the traffic stream after a service node is approximated by

$$c_{serve}^2 = (1 - \rho^2)\, c_{in}^2 + \rho^2 \max(c_S^2, 0.2) . \tag{5.6}$$

It is a combination of the squared coefficients of variation of the service time (c_S^2) and the interarrival time, weighted by the square of the utilization.

The traffic rate after merging (λ_{merge}) is the sum of the rates of the contributing traffic streams (λ_{in}). In our model, only merging of identical traffic streams occurs, so that it suffices to use

$$\lambda_{merge} = k\lambda_{in} , \tag{5.7}$$

where k denotes the number of merged traffic streams. For c^2_{merge}, the squared coefficient of variation after merging k identical traffic streams the following approximation can be derived from the one used in QNA:

$$c^2_{merge} = (1-w) + wc^2_{in} , \tag{5.8}$$

where

$$w = \frac{1}{1 + 4(1-\rho)^2(k-1)} . \tag{5.9}$$

In this expression, ρ is the utilization of the service node the traffic stream is going to.

After splitting a traffic stream into k identical traffic streams, the traffic rate (λ_{split}) and the squared coefficient of variation (c^2_{split}) of an outgoing stream are given by

$$\lambda_{split} = \frac{1}{k}\lambda_{in} , \text{and} \tag{5.10}$$

$$c^2_{split} = \frac{1}{k}c^2_{in} + \left(1 - \frac{1}{k}\right) . \tag{5.11}$$

For step 2, the calculation of performance measures per node, the following expressions are used. The mean delay experienced by a customer in a service node ($E[T]$), is defined as the sum of the mean waiting time ($E[W]$) and the mean service time in the node, i.e.,

$$E[T] = E[W] + \frac{1}{\mu} . \tag{5.12}$$

The mean waiting time is approximated as

$$E[W] = \frac{1}{\mu}\frac{\rho}{(1-\rho)}\frac{\left(c^2_{in} + c^2_S\right)}{2}g , \tag{5.13}$$

where g is defined as follows:

$$g = \begin{cases} e^{-\frac{2(1-\rho)}{3\rho}\frac{\left(1-c_{in}^2\right)^2}{c_{in}^2+c_S^2}} & , \; c_{in}^2 < 1 \; , \\ 1 & , \; c_{in}^2 \geq 1 \; . \end{cases} \tag{5.14}$$

Note that Eq. 5.13 is exact for M/M/1 and M/G/1 queues, where g equals 1.

Finally, we give one of the expressions that is used in step 3. It expresses the mean sojourn time ($E[T_{total}]$) experienced by a customer routed through a series of nodes with mean delay $E[T_j]$ as

$$E[T_{total}] = \sum_j E[T_j] \; . \tag{5.15}$$

5.3.2 Analysis of Reassembly and Segmentation Nodes

In Section 5.2 we have introduced the notion of reassembly and segmentation nodes, and used them in the modelling of a CLS. In order to enable us to analyse this model using the techniques of QNA, we have developed expressions for the first two steps of the QNA analysis applied to the new types of nodes in [54] and [56]. These expressions will be described now.

The traffic rate of a stream departing from a reassembly node (λ_{reass}) is given by

$$\lambda_{reass} = \frac{\lambda_{in}}{l} \; . \tag{5.16}$$

Recall that λ_{in} is the arrival rate to the node, and that l is the average number of arriving customers that is combined into a single departing customer. The squared coefficient of variation of the departing traffic stream is as follows:

$$c_{reass}^2 = \frac{c_{in}^2}{l} + c_L^2 \; , \tag{5.17}$$

where c_L^2 is the squared coefficient of variation of L, the random variable describing the number of arriving customers to be combined for a departure. Finally, the mean time spent by a customer in a reassembly node, $E[T_{reass}]$, is given by

$$E[T_{\text{reass}}] = \frac{\left(c_L^2 + 1\right) l - 1}{2\lambda_{\text{in}}} \ . \tag{5.18}$$

These expressions for a reassembly node (Eq. 5.16, Eq. 5.17, and Eq. 5.18) are exact, given that the consecutive interarrival times of customers are i.i.d., as is assumed in QNA.

A segmentation node is always analysed in conjunction with a service node to which customers are proceeding after leaving the segmentation node. The reason for this is that analysing the nodes separately yields inaccurate results for the mean waiting time in the service node. The combination of nodes can be looked at as a service node with batch arrivals, i.e., batches arrive with a rate λ_{in}, and their interarrival time distribution has a squared coefficient of variation c_{in}^2.The service time that is needed to serve an entire batch relates to the service time of a single customer as follows. The rate at which batches are served, μ_B, is defined as

$$\mu_B = \frac{\mu}{l} \ , \tag{5.19}$$

where l is the mean of L, the batch size, or the number of customers to depart from the segmentation node upon arrival of a customer. The squared coefficient of variation of the service time, $c_{S_B}^2$, is defined as

$$c_{S_B}^2 = \frac{c_S^2}{l} + c_L^2 \ , \tag{5.20}$$

where c_L^2 is the squared coefficient of variation of L. Consequently, the utilization of the combined node, ρ, is given by

$$\rho = \frac{\lambda_{\text{in}}}{\mu_B} \ . \tag{5.21}$$

The rate (λ_{bserve}) and squared coefficient of variation (c_{bserve}^2) of the traffic stream departing from the combined segmentation and service node have been derived in [54] as follows:

$$\lambda_{\text{bserve}} = \lambda_{\text{in}} l \ , \text{and} \tag{5.22}$$

$$c_{\text{bserve}}^2 = \rho c_S^2 + (1-\rho^2)\, c_{\text{in}}^2 l + (1-\rho)^2 (l-1) \ . \tag{5.23}$$

The waiting time of a customer arriving in a batch at the node (W_{bserve}) can be seen as consisting of two parts. The first part (W_B) is the waiting time between the arrival of the batch and the start of service for the first customer in the batch. The second part (W_C) is the waiting time from the start of service for the first customer in the batch until the start of service for the concerned customer. Consequently the mean waiting time of a single customer can be expressed as

$$E[W_{\text{bserve}}] = E[W_B] + E[W_C] \ . \tag{5.24}$$

The mean waiting time until the start of service of the first customer in the batch can be found by using Eq. 5.13 and replacing the service rate μ by μ_B, and the squared coefficient of variation of the service time c_S^2 by $c_{S_B}^2$:

$$E[W_B] = \frac{1}{\mu_B}\frac{\rho}{(1-\rho)}\frac{\left(c_{\text{in}}^2 + c_{S_B}^2\right)}{2}g \ . \tag{5.25}$$

Finally, it has been derived in [54] that the second part of Eq. 5.24 can be expressed as

$$E[W_C] = \frac{\left(c_L^2 + 1\right)l - 1}{2\mu} \ . \tag{5.26}$$

Most of the expressions for the analysis of a segmentation node in conjunction with a following service node are exact. However, for g in Eq. 5.25, we use the same approximation as in QNA (Eq. 5.14), which is exact if the arrival process to the segmentation node is a Poisson process. Furthermore, for the derivation of Eq. 5.23, an approximate step has been used, which has also been used in QNA.

Now that we have the required expressions for reassembly nodes and segmentation nodes, we are able to analyse queueing networks containing these nodes using the techniques applied in QNA. In the following subsections we will apply some of the techniques to find an explicit expression for the mean sojourn time of cells in a CLS. First, we will determine the rate and the squared coefficient of variation of the departing traffic stream, and the mean sojourn time of each submodel.

5.3.3 The Arrival Process

In Section 5.2.3, we have defined the arrival process to a single CLSM as a Bernoulli process with arrival probability p_{arrival}. As a result, the interarrival time

distribution has a geometric distribution. The cell arrival rate to the cell input processing submodel (λ_{arrival}) equals the arrival probability, i.e.,

$$\lambda_{\text{arrival}} = p_{\text{arrival}} = m\rho_c\nu \ . \tag{5.27}$$

The squared coefficient of variation of the interarrival time distribution (c^2_{arrival}) can be derived from a known result for the geometric distribution ([84]):

$$c^2_{\text{arrival}} = 1 - p_{\text{arrival}} \ . \tag{5.28}$$

5.3.4 Cell Input Processing

For the cell input processing submodel, it is very simple to derive the rate (λ_{cip}) and squared coefficient of variation (c^2_{cip}) of the departing traffic stream. Since the submodel only involves a fixed delay, the traffic flow is not changed, i.e.,

$$\lambda_{\text{cip}} = \lambda_{\text{arrival}} \text{ , and} \tag{5.29}$$

$$c^2_{\text{cip}} = c^2_{\text{arrival}} \ . \tag{5.30}$$

Furthermore, the (mean) delay of the submodel is by definition

$$E[T_{\text{cip}}] = \frac{1}{\mu_{\text{cip}}} \ . \tag{5.31}$$

5.3.5 Buffering

We can find the rate of the traffic streams departing from the buffering submodel (λ_{buf}) by applying successively Eq. 5.10, Eq. 5.16, and Eq. 5.10 again. The incoming traffic stream is split into m streams with equal probabilities, reassembled according to L, and split into k streams with equal probabilities, i.e.,

$$\lambda_{\text{buf}} = \frac{\lambda_{\text{cip}}}{klm} \ . \tag{5.32}$$

The squared coefficient of variation of the departing traffic streams (c^2_{buf}) is obtained by successively applying Eq. 5.11, Eq. 5.17, and Eq. 5.11 again:

$$c^2_{\text{buf}} = \frac{1}{k}\left(\frac{1}{l}\left(\frac{1}{m}c^2_{\text{cip}} + 1 - \frac{1}{m}\right) + c^2_L\right) + 1 - \frac{1}{k} = \frac{c^2_{\text{cip}}}{klm} + \frac{c^2_L}{k} + \frac{lm\,(k-1)\ + m - 1}{klm} \ . \tag{5.33}$$

For determining the mean delay experienced by a customer in the buffering submodel ($E[T_{buf}]$), we have to distinguish between message mode and streaming mode of operation. The delay comes from the reassembly node, where customers wait for their (virtual) reassembly.

Message Mode. In message mode, the reassembly is real, i.e., arriving customers have to wait until the last customer for their departing customer (representing the packet they belong to) has arrived. This is also the behaviour that is assumed in [54] for deriving Eq. 5.18. Hence, we can use this expression for the buffering delay in message mode, provided that we substitute λ_{cip}/m for the arrival rate λ_{in} to the reassembly node:

$$E[T_{buf}]_{message} = \frac{m\left(\left(c_L^2 + 1\right)l - 1\right)}{2\lambda_{cip}} . \tag{5.34}$$

Streaming Mode. For streaming mode, the operation of the reassembly node has been defined differently. No real reassembly takes place. All arriving customers are delayed for exactly b cell times. However, the outgoing traffic stream is modelled such that only the first customer of a group (packet) really departs, representing the stream of departures for the group the customer belongs to. The reverse process is performed in the segmentation node, and will be discussed later. This transformation can be done because no service nodes are visited between the reassembly node and the segmentation node. By definition, the buffering delay in streaming mode is given by:

$$E[T_{buf}]_{stream} = b . \tag{5.35}$$

5.3.6 Traffic Shaping

In order to analyse the traffic shaping submodel, we first derive the rate (λ_{s_in}) and squared coefficient of variation ($c_{s\_in}^2$) of the traffic stream entering a segmentation node. These can be derived using Eq. 5.7 and Eq. 5.8:

$$\lambda_{s\_in} = k\lambda_{buf} \text{ , and} \tag{5.36}$$

$$c_{s\_in}^2 = (1 - w) + wc_{buf}^2 \text{ , where} \tag{5.37}$$

$$w = \frac{1}{1 + 4(1-\rho)^2(k-1)} . \tag{5.38}$$

Here, ρ is the utilization of the service node following the segmentation node, and is defined as follows:

$$\rho = \frac{l\lambda_{s\_in}}{\nu} = \rho_c \ . \tag{5.39}$$

The characterization of the departure process and the mean delay for the traffic shaping submodel is treated separately for the message mode and the streaming mode of operation.

Message Mode. Let us first derive the rate (λ_{ts}) and squared coefficient of variation (c_{ts}^2) of the traffic stream departing from the traffic shaping submodel, i.e., from the service node following the segmentation node. In Section 5.3.2, it has been argued that the two nodes should be analysed in conjunction, using a batch arrival approach. We use Eq. 5.22 and Eq. 5.23 to obtain

$$\lambda_{ts} = l\lambda_{s\_in} \text{ , and} \tag{5.40}$$

$$c_{ts}^2 = (1-\rho^2)\, c_{s\_in}^2 l + (1-\rho)^2 (l-1) \ , \tag{5.41}$$

with ρ as in Eq. 5.39.

The only node causing delay in the traffic shaping submodel is the service node modelling the traffic shaper. In Section 5.2.6, we found that the traffic shaper can be modelled by such a service node, if the service time is not accounted for in the total delay. Consequently, the mean delay ($E[T_{ts}]_{message}$) equals the mean waiting time, which can be expressed as in Eq. 5.24:

$$E[T_{ts}]_{message} = E[W_{tsB}] + E[W_{tsC}] \ . \tag{5.42}$$

In order to define the components of this batch arrival approximation, we need to characterize the service time of a batch of customers. Since the service node has been defined to have a service rate ν, the rate at which batches are served can be derived from Eq. 5.19 as

$$\mu_B = \frac{\nu}{l} \ . \tag{5.43}$$

Furthermore, since the service time of a single customer is deterministic, i.e., its squared coefficient of variation equals zero, the squared coefficient of variation of the service time of a batch of customers can be derived from Eq. 5.20 as

$$c_{S_B}^2 = c_L^2 \ . \tag{5.44}$$

Now we are able to define the components of the mean delay of the traffic shaper in case of message mode. The mean waiting time until the start of service for the first customer in the batch can be obtained from Eq. 5.25, substituting Eq. 5.43 and Eq. 5.44:

$$E[W_{\mathrm{ts}B}] = \frac{l}{\nu}\frac{\rho}{1-\rho}\frac{c_{\mathrm{s\_in}}^2 + c_L^2}{2} g \ , \text{ where} \tag{5.45}$$

$$g = \begin{cases} e^{-\frac{2(1-\rho)}{3\rho}\frac{\left(1-c_{\mathrm{s\_in}}^2\right)^2}{c_{\mathrm{s\_in}}^2 + c_L^2}} & , \ c_{\mathrm{s\_in}}^2 < 1 \ , \\ 1 & , \ c_{\mathrm{s\_in}}^2 \geq 1 \ . \end{cases} \tag{5.46}$$

The mean waiting time of a customer after the first customer in the batch has commenced service can be obtained from Eq. 5.26:

$$E[W_{\mathrm{ts}C}] = \frac{\left(c_L^2 + 1\right)l - 1}{2\nu} \ . \tag{5.47}$$

Streaming Mode. For the streaming mode of operation, things are more complicated. Recall that in this case a customer entering the traffic shaping submodel actually represents a sequence of customers that have been combined into a single customer in the reassembly node. The average of the number of customers in the sequence is given by l, and its squared coefficient of variation is given by c_L^2. The successive interdeparture times of the customers in the sequence from the segmentation node equal their interarrival times at the reassembly node. As a consequence, the mean intercell time of this sequence ($1/\lambda_{\mathrm{seq}}$) can be characterized using Eq. 5.10 and Eq. 5.29 as

$$\lambda_{\mathrm{seq}} = \frac{\lambda_{\mathrm{cip}}}{m} = \frac{\lambda_{\mathrm{arrival}}}{m} \ , \tag{5.48}$$

because the traffic stream departing from the cell input processing submodel is split over m streams before it arrives at the reassembly node. Note that λ_{seq} does not characterize a traffic stream, but only the sequence of customers that is represented by a single customer entering the traffic shaping submodel.

For the departure process from the service node, we approximate the sequence of departures from the segmentation node as if all customers depart simultaneously, i.e., we use the approximations that are used for message mode (Eq. 5.40 and Eq. 5.41). Note that there will be no differences between the departure process in message and in streaming mode as long as the interdeparture times of the customers in a sequence created by the segmentation node are smaller than the service time, because in that case the service time will determine the departure time of a customer of the sequence.

In order to approximate the mean waiting time of a customer in streaming mode, we use again the fact that customers arrive in batches at the service node. Now, the customers in the batch do not arrive simultaneously, as in message mode, but with a small interarrival time (with mean $1/\lambda_{seq}$). Let W_{first} be the waiting time of the first customer of a batch (or sequence). Let t_a be the difference in arrival time between the first and an arbitrary customer in the sequence. Finally, let t_s be the difference in the time service starts between the first and the same arbitrary customer. In Figure 5-13, it can easily be seen that the time between the arrival of the first customer and start of service of an arbitrary (in Figure 5-13, the fourth) customer equals either $W_{first} + t_s$ or $t_a + W$, where W is the waiting time of the arbitrary customer. Consequently, we can define the waiting time of an arbitrary customer by

$$W = W_{first} + t_s - t_a \ , \tag{5.49}$$

and its expected value by

$$E[W] = E[W_{first}] + E[t_s] - E[t_a] \ . \tag{5.50}$$

The mean waiting time of the first customer of a sequence is approximated as if the server had real batch arrivals, i.e., using Eq. 5.45 we get:

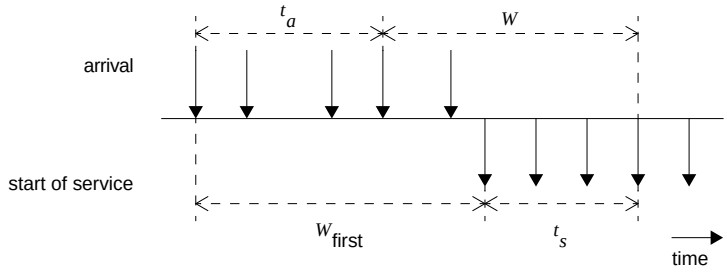


Figure 5-13: Waiting Time of an Arbitrary Customer in Streaming Mode

$$E[W_{\text{first}}] = E[W_{\text{ts}B}] = \frac{l}{\nu}\frac{\rho}{1-\rho}\frac{c_{\text{s\_in}}^2 + c_L^2}{2}g \ . \tag{5.51}$$

For the calculation of $E[t_a]$, we use an expression similar to Eq. 5.47, which has been derived using Eq. 164 of [144]:

$$E[t_a] = \frac{\left(c_L^2 + 1\right)l - 1}{2\lambda_{\text{seq}}} \ . \tag{5.52}$$

Recall that $1/\lambda_{\text{seq}}$ is the expected interarrival time of the sequence. The remaining factor in the expression defines how many interarrival times are expected to expire until the departure of an arbitrary customer, e.g., $(l-1)/2$ if the length of the sequence is always l customers (i.e., $c_L^2 = 0$).

For $E[t_s]$, we use a simple approximation. Let σ be the probability that the first customer of a sequence finds the server busy. We assume that the server stays busy during the arrival of the other customers of the sequence. In this case, t_s is determined by the service time of the preceding customers in the sequence. We use the result from [144] again to obtain:

$$E[t_s \,|\, \text{server busy}] = \frac{\left(c_L^2 + 1\right)l - 1}{2\nu} \ . \tag{5.53}$$

We assume that otherwise, if the first customer finds the server empty, with probability $1-\sigma$, all customers of the sequence are served immediately upon arrival, i.e., $t_s = t_a$. Consequently,

$$E[t_s \,|\, \text{server empty}] = E[t_a] \ . \tag{5.54}$$

Concluding, we use the following approximation for $E[t_s]$:

$$E[t_s] = \sigma\frac{\left(c_L^2 + 1\right)l - 1}{2\nu} + (1-\sigma)\,E[t_a] \ . \tag{5.55}$$

This results in the following expression for $E[W]$:

$$E[W] = E[W_{\text{first}}] + \sigma\left(\frac{\left(c_L^2 + 1\right)l - 1}{2\nu} - \frac{\left(c_L^2 + 1\right)l - 1}{2\lambda_{\text{seq}}}\right) . \tag{5.56}$$

Using Eq. 5.27 and Eq. 5.48, this can be rewritten as:

$$E[W] = E[W_{\text{first}}] - \sigma \frac{1-\rho_c}{\rho_c} \frac{\left(c_L^2 + 1\right) l - 1}{2\nu} \ . \tag{5.57}$$

We approximate the probability that the first customer of a sequence finds the server busy, by

$$\sigma = \rho \ . \tag{5.58}$$

This expression is exact if these first customers arrive according to a Poisson process, because Poisson arrivals see time averages (PASTA).

Now we are able to formulate the expression for the mean delay in the traffic shaping submodel in streaming mode ($E[T_{\text{ts}}]_{\text{stream}}$). Similar to the message mode case, it equals the mean waiting time in the service node, i.e.,

$$E[T_{\text{ts}}]_{\text{stream}} = E[W] \ , \tag{5.59}$$

with $E[W]$ as in Eq. 5.57.

5.3.7 Cell Output Processing

For the cell output processing submodel, it is straightforward to derive an expression for the mean delay. Let us first characterize the arrival process to the service node using Eq. 5.7, Eq. 5.8 and Eq. 5.9:

$$\lambda_{\text{srv\_in}} = m\lambda_{\text{ts}} \text{ , and} \tag{5.60}$$

$$c^2_{\text{srv\_in}} = (1-w) + wc^2_{\text{ts}} \text{ , where} \tag{5.61}$$

$$w = \frac{1}{1 + 4(1-\rho)^2 (m-1)} \ . \tag{5.62}$$

Since the service time of the service node is always one cell time ($\mu_{\text{cop}} = 1$), we obtain for the utilization:

$$\rho = \lambda_{\text{srv\_in}} \ . \tag{5.63}$$

Furthermore, since the service time is fixed,

$$c_S^2 = 0 \ . \tag{5.64}$$

For the mean delay a customer experiences in the cell output processing submodel $E[T_{cop}]$ we use Eq. 5.12 and Eq. 5.13:

$$\mathrm{E}[T_{cop}] = \frac{\rho}{1-\rho}\frac{c_{srv\_in}^2}{2}g + 1 \text{ , with} \tag{5.65}$$

$$g = \begin{cases} e^{-\frac{2(1-\rho)}{3\rho}\frac{\left(1-c_{srv\_in}^2\right)^2}{c_{srv\_in}^2}} & , \; c_{srv\_in}^2 < 1 \text{ ,} \\ 1 & , \; c_{srv\_in}^2 \geq 1 \text{ .} \end{cases} \tag{5.66}$$

5.3.8 Network Analysis

Having obtained approximations for the mean delay, experienced by an arbitrary customer in each of the submodels, it is easy to derive the mean sojourn time in a CLS. Since a customer goes to all types of submodels once, the mean sojourn time is the sum of the mean sojourn times of the submodels, i.e., using Eq. 5.15, we get for the mean sojourn time in message mode ($E[T]_{message}$):

$$E[T]_{message} = E[T_{cip}] + E[T_{buf}]_{message} + E[T_{ts}]_{message} + E[T_{cop}] \text{ .} \tag{5.67}$$

Similarly, we get for the mean sojourn time in streaming mode ($E[T]_{stream}$), using Eq. 5.15 again:

$$E[T]_{stream} = E[T_{cip}] + E[T_{buf}]_{stream} + E[T_{ts}]_{stream} + E[T_{cop}] \text{ .} \tag{5.68}$$

Table 5-4 summarizes the component parts of both $E[T]_{message}$ and $E[T]_{stream}$. It can be seen that the difference between the two stems partly from the difference in buffering. In message mode, customers experience delay for reassembly, which is for instance depending on the packet length. In streaming mode, customers are delayed for a fixed time in the FIFO buffer. The other part of the difference stems from the traffic shaper, where the delay is larger in message mode, because all customers of a packet arrive simultaneously, rather than with a small interarrival time.

	Message Mode	Streaming Mode
$E[T_{cip}]$	$\frac{1}{\mu_{cip}}$	$\frac{1}{\mu_{cip}}$
$E[T_{buf}]$	$\frac{m\left(\left(c_L^2+1\right)l-1\right)}{2\lambda_{cip}}$	b
$E[T_{ts}]$	$\frac{l}{\nu}\frac{\rho}{1-\rho}\frac{c_{s\_in}^2+c_L^2}{2}g+\frac{\left(c_L^2+1\right)l-1}{2\nu}$	$\frac{l}{\nu}\frac{\rho}{1-\rho}\frac{c_{s\_in}^2+c_L^2}{2}g-\sigma\frac{1-\rho_c}{\rho_c}\frac{\left(\left(c_L^2+1\right)l-1\right)}{2\nu}$
$E[T_{cop}]$	$\frac{\rho}{1-\rho}\frac{c_{srv\_in}^2}{2}g+1$	$\frac{\rho}{1-\rho}\frac{c_{srv\_in}^2}{2}g+1$

Table 5-4: Delay Components

5.4 Evaluation

In the previous section, a set of expressions has been presented, which allows us to approximate the mean delay, experienced by a cell in a CLS. This set of expressions can be considered as a tool for performance predictions. The purpose of this section is to test the accuracy of the tool, by comparing the results from the approximations with results from simulations, and to use the tool to compare different alternatives for the design of a CLS.

First, in Section 5.4.1, we briefly describe the simulation model that has been used to test the approximations. In Section 5.4.2 we describe the default values that have been used for system and workload parameters. Then we describe a number of experiments, performed with the analysis tool. All experiments are done for Class A as well as Class B implementation architectures, and for message as well as streaming mode of operation. First, in Section 5.4.3, we experiment with the utilization of the connections between CLSs. In Section 5.4.4 we experiment with the load on a CLS, and in Section 5.4.5 we experiment with the length of the packets that are routed by the CLS. Finally, in Section 5.4.6, draw some general conclusions, and answer the questions posed in Section 5.1.

5.4.1 Simulation Model

In order to check the accuracy of the results obtained with the approximate expressions, a simulation model has been developed. This model is documented in [59], in which also the source code of the model, which can be executed with the modelling package QNAP2 ([108]), can be found. Basically, the simulation model corresponds to the model that has been introduced in Section 5.2. However, a number of simplifications that have been made in order to allow for the approximate analysis, are not necessary for the simulation program. Let us now discuss these simplifications.

In the approximate model, a group of customers representing the cells of a packet was combined into a single customer representing the packet, in the reassembly node. In the segmentation node, from this customer, a group of customers was created according to the same distribution as used for the reassembly. For a single packet, the number of customers that are combined, and the number of customers that are created are independent, and thus not necessarily the same. Clearly, these two numbers are equal in a real system, since they both represent the number of cells a certain packet is contained in. In the simulation model we do not perform the combination and creation of customers. Instead, we forward all customers from the reassembly directly to the traffic shaping node. Thus we avoid the simplification made in the approximate analysis. One of the reasons for combining a group of customers was that they should all be routed equally. This is also accommodated for in the simulation model.

In the model, presented in Section 5.2, the packet processing (see e.g., Figure 5-1) has not been taken into account, since it is performed simultaneously with the buffering, and is assumed to be completed earlier. In the simulation model for the message mode of operation, packet processing is modelled as well. It is modelled by a series of two single server queues. The first one, which models the routing function, has an Erlang-10 service time distribution with a mean of $1/0.27$ cell times. Thus, it can serve 100 000 packets per second, which is the target value for a routing module described in [28]. The second one, which models the other packet processing functions, has an Erlang-10 service time distribution with a mean of $1/0.81$ cell times. It can serve 300 000 packets per second, which is also the target value in [28]. In the simulation model, customers are only forwarded to the traffic shaping submodel if the buffering as well as the packet processing have been completed.

In streaming mode of operation, customers arriving in the reassembly node simply wait for b slot times, and are then forwarded to the traffic shaping submodel, assuming that the packet processing has been completed in the mean time. All customers representing cells of the same packet are routed to the same traffic shaper.

The simulation results shown in this section are the 95% confidence intervals of the estimated measure. Simulation times have been taken such that the confidence intervals are sufficiently small.

5.4.2 Parameter Values

For the experiments of the following subsections, we have to assume certain values for the parameters of the model. Here, we give the default values for the parameters. During the experiments, some of the model parameters are varied. The parameters that are not explicitly mentioned in the concerned subsection have the value as indicated here.

Note that the unit of time is still the time needed to transmit a single cell on an ATM link, i.e., 2.7 μs on a 155.520 Mbits/s link. This unit of time is also used to express the mean delay of cells experienced in the CLS.

Let us first discuss the system parameters (Table 5-5). We consider a moderate size CLS. During all the experiments, we assume that the CLS that is modelled consists of 5 CLSMs in case of a Class A architecture, or of 5 Input CLSMs and 5 Output CLSMs in case of a Class B architecture ($n = 5$). Furthermore, we assume that the CLS has connections to 50 other CLSs or end-systems ($k = 50$). This implies that the number of outgoing and incoming connections per CLSM (m) is 50 for a Class A architecture, and 10 for a Class B architecture. In the approximate model, analysed in Section 5.3, the value of n does not play a role, other than through k and m, via Eq. 5.2.

We further assume that the cell input processing and the cell output processing can be performed in a single time-slot, i.e., μ_{cip} and μ_{cop} are both 1. The effect of this assumption on the results is marginal, since the major delay components stem from the buffering and traffic shaping submodels. The size of the FIFO buffer in streaming mode is assumed to be 700 cells, which is the size aimed at in the CLS described in [28]. Thus, the delay introduced by this buffer (b) equals 700 cell times.

parameter	Class A	Class B
n	5	5
k	50	50
m	50	10
μ_{cip}	1	1
μ_{cop}	1	1
b	700	700
ν_{total}	0.1	0.1

Table 5-5: Default Values for System Parameters

To facilitate the evaluation we define a new parameter, ν_{total}, which represents the total reserved bandwidth on all outgoing connections to a single adjacent CLS or end-system, and consequently on all incoming connections from a single adjacent CLS or end-system. Recall that ν denotes the peak cell rate of a single connection, which corresponds to the fraction of the link bandwidth reserved for this connection. Let ν_{total} be defined as the sum of the peak cell rates for all connections to or from a single adjacent node, i.e., the maximum rate at which cells can be transferred between two adjacent CLSs. The following relation holds:

$$\nu_{total} = \frac{nm}{k}\nu \ , \tag{5.69}$$

since the number of connections per adjacent node equals the total number of connections originating in a CLS (nm), divided by the number of adjacent nodes that should be reached (k). From Eq. 5.1 and Eq. 5.2, it can easily be seen that $\nu_{total} = n\nu$ for a Class A architecture, and $\nu_{total} = \nu$ for a Class B architecture. This is exactly how the two classes have been defined.

In order to allow for a fair comparison of Class A and Class B architectures, we use the same value of ν_{total} for both classes during the experiments. The default value is such that the total link capacity is reserved, i.e., $\nu_{total} = n/k = 0.1$.

parameter	value
ρ_c	0.8
l	10
c_L^2	0.9

Table 5-6: Default Values for Traffic Parameters

The default parameters for the traffic arriving in a CLS are as follows (Table 5-6). Experiments have suggested a connection utilization of 0.8 (Section 5.4.3). We assume a mean packet length of 10 cells. This value is based on measurements in [43]. Furthermore, we assume that the packet length is geometrically distributed. This leads to a squared coefficient of variation of the packet length distribution of 0.9 ([84]).

In the experiments that follow, we distinguish four different systems. The different results are identified as follows. The results for a Class A implementation architecture (multiple connections to the next node) are labelled '$A_{message}$' if the CLS operates in message mode and '$A_{streaming}$' if the CLS operates in streaming mode. Similarly, the results for a Class B implementation architecture (single connection to the next node) are labelled '$B_{message}$' and '$B_{streaming}$' respectively.

5.4.3 Experiment 1: Connection Utilization

The first experiment to be discussed is to vary the connection utilization (ρ_c). The purpose of the experiment is to investigate what value must be used for the peak cell rate parameter (ν) in the traffic shaper, given a certain load on the CLSM (expressed by the arrival probability, $p_{arrival}$).

During the experiment, we vary the connection utilization and the peak cell rate parameter in such a way that their product ($\rho_c \nu = p_{arrival}/m$, using Eq. 5.3) remains constant. We keep $p_{arrival}$ equal to 0.5, so that ρ_c equals 0.5, if we reserve the total link capacity for the outgoing connections, i.e., if $\nu = 1/m$. By reserving less, we can investigate the delay of the CLS for connection utilizations larger than 0.5.

Note that for a given connection utilization, Class A and Class B architectures use different peak cell rates for the traffic shaping, because of the different number of outgoing connections. However, the sum of the peak cell rates of all connections to a single adjacent node (ν_{total}) is the same for both classes.

The results from the experiment (Figure 5-14) show that the mean delay experienced by cells in a CLS is higher for a Class A architecture than for a Class B architecture. This effect is partly explained by the difference in peak cell rate, since in general, a single server gives a lower delay than a number of parallel servers with the same total capacity. For the message mode of operation, there is an additional explanation for the difference in delay between Class A and B. For that mode of

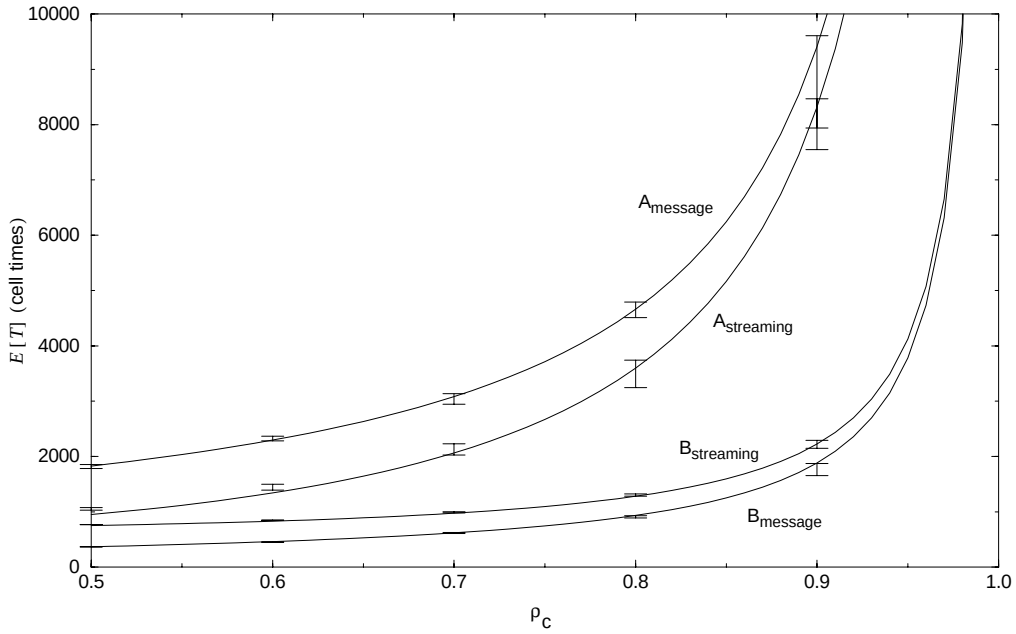


Figure 5-14: Delay with Varying Connection Utilization and Fixed Load

operation, Class B results in a lower reassembly delay, because the cells of a packet arrive at a higher speed on the single connection with a high peak cell rate.

The difference between the delay of a CLS operating in message mode and a CLS operating in streaming mode is mainly explained by the time needed for buffering. In streaming mode, this is a fixed delay (b). In message mode, it is the reassembly delay, which depends on the rate at which the cells of a packet arrive. For Class B, this rate is relatively high, because a packet is arriving on a connection with a high bandwidth assigned to it. Hence the delay is low, lower than in streaming mode. For Class A, the rate is relatively low, because a packet is arriving on one of the connections that has part of the total bandwidth (ν_{total}) assigned to it. Hence, the delay is high.

The effect of an increase in connection utilization is similar for the four systems, although the relative difference between message and streaming mode disappears for high utilization. The delay increases sharply above $\rho_c = 0.8$.

For a certain outgoing connection, the value of the peak cell rate parameter has to be agreed on in the traffic contract. The eventual value that will be chosen, i.e., the bandwidth that is reserved on the connection, will depend on the bandwidth cost, the traffic load, and the allowed delay. In general, it can be concluded that it seems prudent to choose such a peak cell rate value that the connection utiliza-

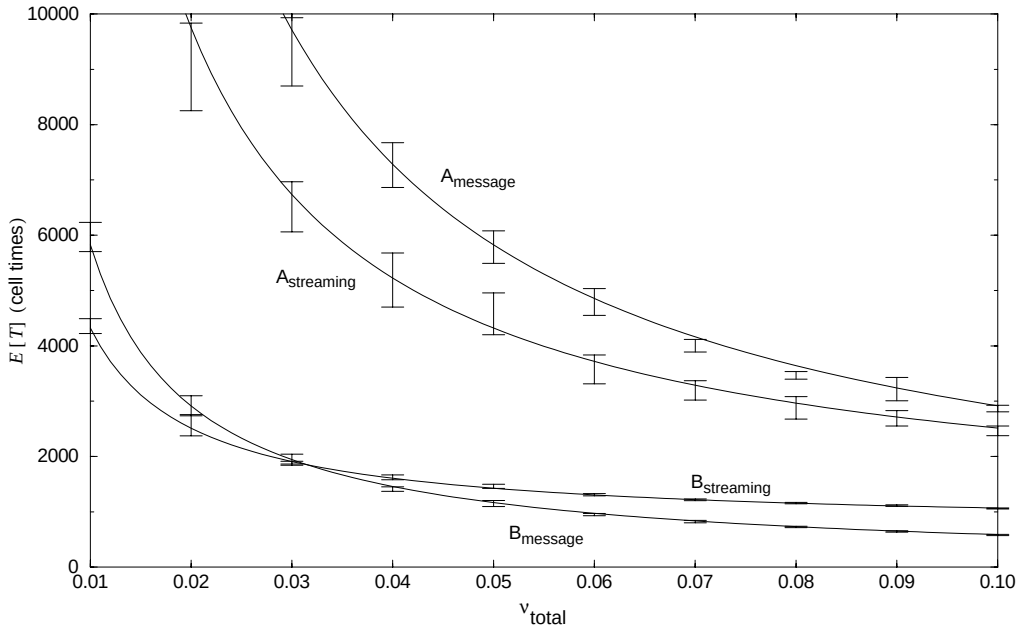


Figure 5-15: Delay with Varying Peak Cell Rate

tion is around 0.8, i.e., to choose $\nu = p_{\text{arrival}} / (m \times 0.8)$ (Eq. 5.3). Reserving more is inefficient and costly, and reserving less leads to a higher connection utilization, and hence significantly higher delays.

5.4.4 Experiment 2: Load on a CLS

In the second experiment, the influence of the load on the CLS is investigated. From Eq. 5.3, it can be observed that the load on a single CLSM, and hence on the entire CLS, can be varied by varying either the peak cell rate of the (incoming) connections (ν), or the utilization of these connections (ρ_c).

In Figure 5-15, the peak cell rate of the connections between two CLSs is varied, while the utilization of the connections remains constant at 0.8. In order to be able to compare the different classes of implementation architectures, we consider ν_{total} instead of ν (see Section 5.4.2).

Again it can be seen that CLSs with a Class B architecture impose a lower delay on the cells than CLSs with a Class A architecture. If the peak cell rate, and hence the load on the CLS, increases, the mean delay decreases. This can be explained by the fact that the service rate of the traffic shaper increases, because of the increased peak cell rate for the outgoing connection. Furthermore, in message

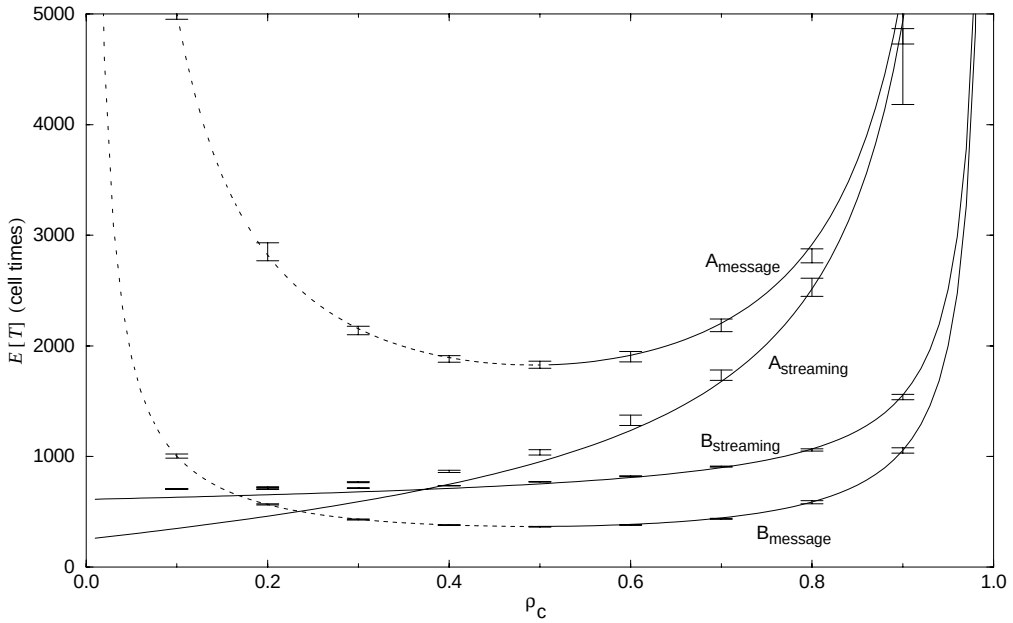


Figure 5-16: Delay with Varying Load

mode the decrease is even sharper, because the reassembly delay decreases at higher rates.

In Figure 5-16, the sum of the peak cell rates on all connections from a single adjacent node (ν_{total}) remains constant at 0.1, i.e., $\nu = 0.02$ for Class A, and $\nu = 0.1$ for Class B (Eq. 5.69). For both classes, the sum of all peak cell rates for one CLSM ($m \times \nu$) equals 1, i.e., one cell arrives per cell time if all connections are fully utilized. In the figure, we vary the load on a CLSM, and hence on a CLS, by varying the connection utilization (ρ_c).

Again, we see the general trend that the delay for a Class B architecture is significantly lower than for a Class A architecture. Furthermore, we can see that for low utilization, the delay increases sharply in message mode. This is due to the increasing reassembly time, because of the low arrival rate. In a real system, this high reassembly delay will probably not be seen, since the originating CLS will transfer the cells of the same packet at a high rate, with higher interpacket times, if the utilization is low. This implies that during low utilization, the offered traffic will have a bursty nature, and that our model does not reflect the system behaviour any more. However, since the CLS should be designed to operate well under high (busy-hour) load, we are mostly interested in its behaviour for high utilization.

If the utilization approaches zero, the mean delay in streaming mode should converge to the delay of the FIFO buffer (b = 700 cell times) plus the delay for cell input and output processing (together 2 cell times). This is also shown by the simulation results. However, it is not the case for the approximation of the streaming mode delay. Clearly, this approximation is not accurate for a low connection utilization. However, the operational region of a CLS is at a much higher utilization, typically around 0.8.

If the utilization approaches 1, the mean delay goes to infinity for all four systems. This behaviour, which is similar to the one in the first experiment, is caused by the traffic shaping function.

5.4.5 Experiment 3: Packet Length Distribution

In the last experiment, the influence of the packet length distribution on the mean delay is investigated. First, we vary the mean and then the variability of the packet length. The load on the CLS is fixed.

In Figure 5-17 we can see the mean delay increasing linearly with the mean packet length (l). For the streaming mode of operation, the increase can be explained by the fact that cells arrive to the traffic shaping function in groups. The larger the mean packet length, the larger the groups in which cells arrive at the traffic shaping function. This more bursty arrival process leads to higher waiting times for the traffic shaping. In Eq. 5.51, it can be observed that the most important component of the waiting time of the traffic shaper, i.e., the waiting time until the first customer of the group is served, is proportionally increasing with l.

The same behaviour can be observed for message mode (see Eq. 5.45), and here the increase in delay is even stronger, because the increasing packet length leads to increasing reassembly delays. In Eq. 5.34, it can be seen that the mean buffering delay in message mode, increases almost proportionally with l. In streaming mode, the mean buffering delay is constant (700 cell times).

For moderate mean packet lengths (10 to 20 cells), we see the same trend as in other figures. Class B architectures have a lower delay than Class A architectures; for Class B, message mode has the lowest delay; and for Class A, streaming mode has the lowest delay. For a small mean packet length, streaming mode always has a higher delay than message mode, because of the fixed size FIFO buffer. For high mean packet size, streaming mode outperforms message mode, because of the high reassembly delays.

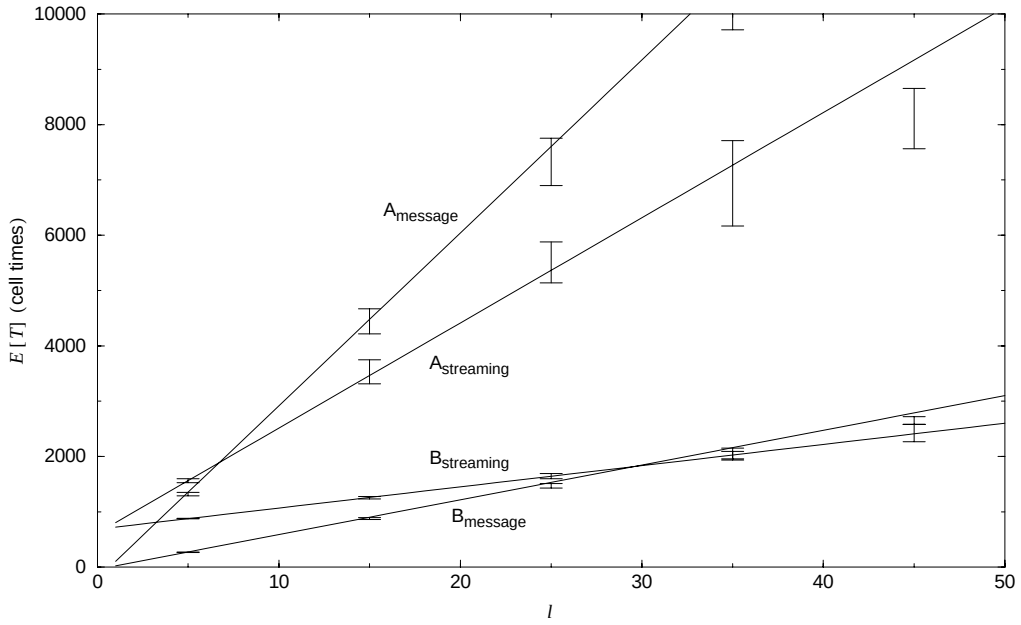


Figure 5-17: Delay with Varying Mean Packet Length

Finally, in Figure 5-18, we vary the variability of the packet length distribution, i.e., we vary its squared coefficient of variation (c_L^2). Note that $c_L^2 = 0$ implies a fixed packet length. We can see that the mean delay is linearly increasing with the variability. The equations mentioned above are linear (not proportional) in c_L^2. A reason for the increasing delay is the higher variability of the size of the batches arriving at the traffic shaper, which results in higher waiting times for the batches. Furthermore, a high variability of the packet length increases the reassembly delay in message mode.

5.4.6 Commentary

The main components of the delay experienced by cells in a CLS are the buffering delay and the traffic shaping delay. In streaming mode, the buffering delay, i.e., the delay of the FIFO buffer, is fixed. In message mode, this delay is heavily influenced by the rate at which the cells of a packet arrive, and the packet length distribution. The traffic shaping delay depends on the connection utilization, the rate at which the traffic shaper works, and on the packet length distribution.

Comparison of the results from the approximate analysis with results from simulations have shown that the approximations are sufficiently accurate. They are mostly within the 95% confidence intervals of the simulation results, which are only a few percent of the value of the measure. Only if the connection utilization

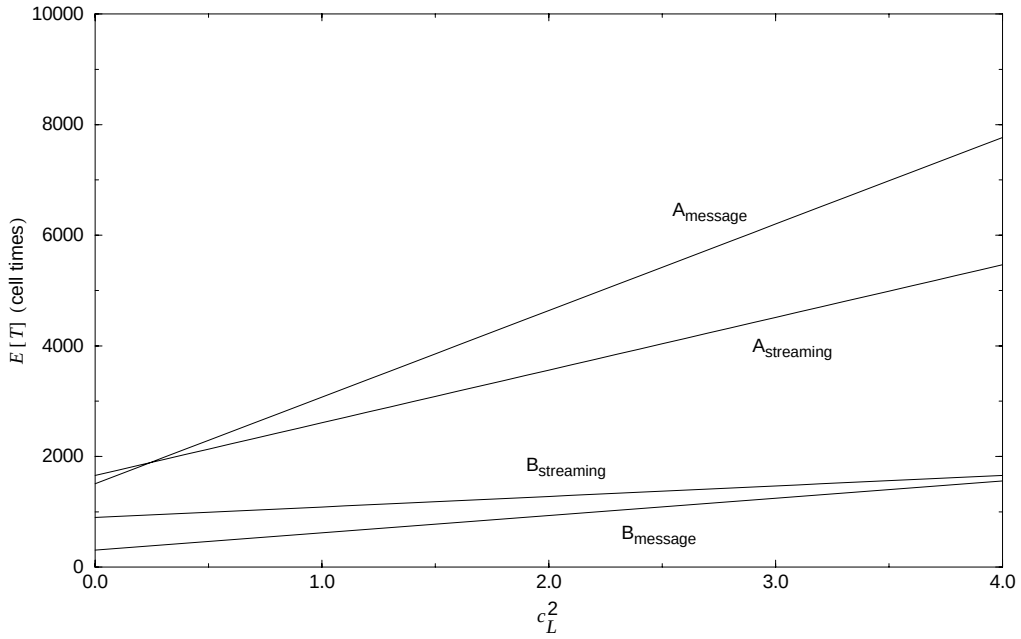


Figure 5-18: Delay with Varying Packet Length Variability

is low, the approximation becomes less accurate. However, this is outside the intended scope of the model.

The approximate analysis provides insight in the behaviour of the system, which is not necessarily provided by simulation. Furthermore, results can be obtained in a split second, while the simulation time for a single confidence interval has an order of magnitude of hours.

Answers to the Research Questions

Let us now return to the questions of Section 5.1. Questions 6 and 7 are concerned with the throughput and delay of cells for the different implementation architectures respectively. We have seen that the mean delay for CLSs with a Class A implementation architecture is always significantly higher than for those with a Class B architecture. This implies that it is better to perform buffering and traffic shaping in different modules, if a low delay is required.

Although we did not take cell loss into account in the analysis, since we assumed infinite buffers, we can make a statement about the throughput. If the buffers of the real system are properly dimensioned, it can be expected that the loss is low, and hence that the throughput is close to the load. However, for parameter values where the mean delay is increasing sharply, it can be expected that it is not

feasible to avoid loss, and the throughput will converge to a maximum. From Experiments 1 and 2, it can be observed that the throughput of a CLS is limited by the connection utilization (ρ_c), not so much by the total load on the system. The mean delay increases sharply if the connection utilization becomes high. For Class A, this occurs around 0.8, for Class B around 0.9.

Concluding, we can state that Class B architectures outperform Class A architectures, as far as delay and throughput is concerned. Implementation Architectures 2 and 4 operate according to Class B. Architectures 3, 5, and 6 operate according to Class A. The models coincide if only a single CLSM is present in the CLS. Hence, Architecture 1 can be modelled by both models. The results for this architecture equal those for Class A, with the difference that ν_{total} is only $1/5$ of the value given here, since the system consists of 1 CLSM instead of 5.

In order to combine the performance advantage of a Class B implementation architecture with the advantages of Architecture 6 that are mentioned in Chapter 4, a new implementation architecture can be envisaged. This architecture is similar to Architecture 6. The difference is that in this new architecture (Implementation Architecture 7), traffic shaping is performed in an Output Module, which already performed the ATM layer processing. As a result buffering and traffic shaping are performed in different modules. A consequence of performing traffic shaping in the Output Module would be that this module needs to be redesigned especially for a CLS. Additional to the regular functionality, functionality for traffic shaping and for the buffering for the traffic shaper needs to be implemented. However, the Output Module can still be used for regular (connection-oriented) ATM traffic, as long as it knows to which connections traffic shaping should be applied, and to which ones not.

Questions 1 and 3 are concerned with the throughput and delay of cells in message and in streaming mode. The comparison between the modes of operation depends heavily on the modelled class of implementation architectures, and on the used parameters. In general, it can be stated that message mode performs better than streaming mode if the bandwidth of a single connection (ν) is high. For Class B architectures with the used parameters, this is the case if the bandwidth of a single connection is more than 3% of the link bandwidth, i.e., in the order of 5 Mbits/s (see Figure 5-15). This value is higher for a larger mean packet length, since the message mode of operation is more sensitive to packet length (see Figure 5-17).

Question 2 is concerned with the interdeparture times of cells within a packet for the different modes of operation. We have not seen a significant difference in the cell departure process between message and streaming mode. In fact, the departing traffic stream from the traffic shaping submodel is approximated by the same rate and squared coefficient of variation for both modes.

Question 5 is about the traffic parameters needed for the outgoing connection. In Section 5.4.3, it has already been concluded that a bandwidth utilization of 0.8 is a good choice. This implies that the peak cell rate parameter should be defined in such a way that $\rho_c = 0.8$.

The only question that has not been considered yet is Question 4 about the size of the reassembly buffer. This will be dealt with in Chapter 6, where a detailed model of the reassembly buffer is presented and analysed.

5.5 Summary and Concluding Remarks

In this chapter, we have obtained insight in the performance aspects of a CLS. We have developed a queueing network model that allows us to analyse the delay experienced by cells in a CLS. For the analysis, we have used and extended an approximate technique that is also used in the software tool QNA.

We have obtained results that showed that CLSs where buffering and traffic shaping is performed in the same module impose a much higher delay on the cells than CLSs where these functions are implemented in separate modules. This has lead to the proposal of an extra implementation architecture, one where traffic shaping is performed in the Output Modules.

It was further shown that it depends heavily on the used parameters, which mode of operation performs best. In general, message mode of operation performs best if high bandwidth connections are used. Streaming mode of operation performs best for lower bandwidths. Furthermore, streaming mode outperforms message mode if the mean packet length is large.

An important part of a CLS in message mode is the reassembly buffer. We have presented an expression for the mean reassembly delay, which is the time spent in this buffer by an average customer. No attention has been paid yet to the dimensioning of the buffer. In the next chapter, we will model and analyse the reassembly buffer in detail, in order to support this dimensioning.

But after all there are such things,
and these are the things
who'll turn your memories back into dreams again.

Rickie Lee Jones - Pirates

Chapter 6
Performance of the Reassembly Buffer

An important component of a CLS operating in message mode is the reassembly buffer. It is used to store arriving cells until all subsequent cells of the packet they belong to have been received. We have already investigated the delay that is experienced by cells in this buffer, i.e., during the reassembly, in the previous chapter. There, we assumed that the probability that a cell or packet had to be discarded, because the reassembly buffer was full, was sufficiently low. In this section we consider the reassembly buffer in more detail, so that we can predict the loss probability of the buffer, given its size. We come up with an analytical tool, that can be used to determine the necessary size of the buffer, given the required loss probability and a characterization of the load on the buffer.

Only when a CLS operates in message mode, it has one or more reassembly buffers. In streaming mode, FIFO buffers are used to store incoming cells while the associated packet processing is being performed. Here, cells are only discarded if the result from the packet processing is not available in time. The exact time needed for packet processing, e.g., for routing, depends on how the relevant processes are implemented. This is beyond the scope of this dissertation, and hence we concentrate on the buffering in message mode, i.e., on the behaviour of a reassembly buffer.

In Section 6.1, we present the modelling and analysis of the reassembly buffer. In Section 6.2, the analysis is validated, and a numerical example of its use is presented. Finally, in Section 6.3, we summarize the results of this chapter and give some concluding remarks.

6.1 Modelling and Analysis

The modelling and analysis in this section is intended to provide a tool for the dimensioning of the reassembly buffer. It can be used to obtain loss figures for a buffer of a certain size under a certain load. Since the loss of a single cell results in

m	number of incoming connections
n	buffer size (in cells)
$1/p$	average packet length (in cells)
r	probability for a certain silent connection to become active in an arbitrary time-slot
s	probability for a certain active connection to become silent in an arbitrary time-slot
p_{arrival}	probability of a cell arrival in a time-slot
$\boldsymbol{D}$	transition matrix of the DTMC modulating the arrival process
$\boldsymbol{T}$	transition matrix of the lumped model
$\boldsymbol{T}^*$	transition matrix of the decomposed model

Table 6-1: Model Parameters

the loss of the whole packet, to which the cell belonged, we aim at the packet loss probability as the measure obtained from the model.

In a CLS, each (Input) CLSM will have a reassembly buffer. The model that we present here represents a single one, i.e., we consider only a single CLSM. As a result, the traffic arriving at the buffer represents the traffic that arrives at a single CLSM.

The notation that is used to describe the model is given in Table 6-1 for later reference. Furthermore, some transition probabilities and transition matrices have superscripts to indicate the arrival of a certain type of cell during the transition. These superscripts are given in Table 6-2.

First, in Section 6.1.1, we discuss the arrival process that is assumed in the analysis. In Section 6.1.2, we present the buffer model and its analysis. In order to obtain results from the model, numerical solution techniques have to be used. In Section 6.1.3, we apply an approximation technique that allows us to get accurate results, much faster than with the original model.

6.1.1 The Arrival Process

We assume that the arrival process to the reassembly buffer is slotted, similar to the modelling in Section 5.2.3. Again, at most one cell arrives in each time-slot. The cell arrival is assumed to be completed at the end of the slot. We only consider the arrival stream at slot boundaries, i.e., we assume discrete time. The cell arrival process that we will describe in this section is also known as a Discrete-time Markovian Arrival Process (D-MAP) ([7]).

0	no cell arrives
1	a cell arrives
an	a cell arrives from one of the *a*ctive connections, it is *n*ot the last one of a packet
al	a cell arrives from one of the *a*ctive connections, it is the *l*ast one of a packet
sn	a cell arrives from one of the *s*ilent connections, it is *n*ot the last one of a packet
sl	a cell arrives from one of the *s*ilent connections, it is the *l*ast one of a packet

Table 6-2: Superscripts used for Transition Probabilities and Transition Matrices

In the analysis of Chapter 5, we were interested in the delay of the system during a high load period. Therefore, we could assume that all connections were active all of the time. Here, we are interested in the probability that a packet gets lost in the long run. Therefore, we cannot restrict the analysis to periods of high load, and we have to take into account that during some periods no customers arrive on a connection, although bandwidth has been reserved. In the following, we assume that a connection is either fully utilized (active), or not utilized at all (silent). In order to simplify the presentation of the analysis, we have assumed that the all bandwidth of an incoming ATM link is reserved for connections, i.e., during periods in which all connections are active, a cell arrives in each time-slot.

The cells arriving on a connection constitute the packets that should be reassembled. We assume that the packet length is geometrically distributed with a mean of $1/p$ cells, i.e., an arriving cell is with probability p the last one of a packet.

The total number of incoming connections in the CLSM is denoted as m. Let the number of active connections at the beginning of time-slot t be defined by the stochastic process $\{A_t, t=0, 1, 2, \ldots, A_t \in \{0, 1, \ldots, m\}\}$. We assume that A_t is a a Discrete-Time Markov Chain (DTMC) with the following transition matrix ($\boldsymbol{D}$):

$$\boldsymbol{D} = [d_{ij}]_{(m+1)\times(m+1)} . \tag{6.1}$$

Clearly, d_{ij} denotes the probability that j connections are active at the end of a slot[1], given that i connections were active at the beginning of the slot. The elements of the transition matrix are defined as follows:

[1] Since the end of a slot coincides with the beginning of the next slot, we use both terms to refer to this slot boundary.

$$\boldsymbol{D} = \begin{bmatrix} 1-mr & mr & 0 & \dots & 0 & 0 & 0 \\ s & 1-(m-1)r-s & (m-1)r & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & . & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & (m-1)s & 1-r-(m-1)s & r \\ 0 & 0 & 0 & \dots & 0 & ms & 1-ms \end{bmatrix} . \quad (6.2)$$

In order to be able to determine the probability of events, such as the arrival of the last cell of a packet, we decompose the transition matrix of the D-MAP into a number of matrices that give the probabilities of transitions combined with the occurrence of certain events. We define d_{ij}^{0} as the probability that a time-slot is not filled and at the end j connections are active, given that at the beginning of the time-slot, i connections were active. Similarly, we define d_{ij}^{1} as the probability that a time-slot is filled with a cell and at the end of the time-slot j connections are active, given that at the beginning of the slot i connections were active. We can also put this in matrix form:

$$\boldsymbol{D}^{0} = \left[d_{ij}^{0}\right]_{(m+1)\times(m+1)} , \text{ and} \quad (6.3)$$

$$\boldsymbol{D}^{1} = \left[d_{ij}^{1}\right]_{(m+1)\times(m+1)} . \quad (6.4)$$

It can easily be seen that

$$\boldsymbol{D} = \boldsymbol{D}^{0} + \boldsymbol{D}^{1} . \quad (6.5)$$

In general, an arriving cell will originate from one of the active connections. However, it could also originate from one of the silent connections that becomes active during the slot. We furthermore distinguish between cells that are the last one of a packet and cells that are not. We thus define matrices

$$\boldsymbol{D}^{al} = \left[d_{ij}^{al}\right]_{(m+1)\times(m+1)} , \text{ and} \quad (6.6)$$

$$\boldsymbol{D}^{an} = \left[d_{ij}^{an}\right]_{(m+1)\times(m+1)} , \quad (6.7)$$

where the (i,j)-th element is the probability that a cell from one of the *a*ctive connections arrives that is (*al*) or is *n*ot (*an*) the *l*ast one of a packet respectively, while the modulating Markov process makes a transition form state i to state j. Furthermore, we define

$$\boldsymbol{D}^{sl} = \left[d_{ij}^{sl}\right]_{(m+1)\times(m+1)} , \text{ and} \quad (6.8)$$

$$\boldsymbol{D}^{sn} = \left[d_{ij}^{sn}\right]_{(m+1)\times(m+1)} , \tag{6.9}$$

where the (i,j) -th element is the probability that a cell from one of the silent connections arrives that is (*sl*) or is *n*ot (*sn*) the *l*ast one of a packet respectively, while the modulating Markov process makes a transition form state i to state j. Clearly, we have

$$\boldsymbol{D}^{1} = \boldsymbol{D}^{al} + \boldsymbol{D}^{an} + \boldsymbol{D}^{sl} + \boldsymbol{D}^{sn} . \tag{6.10}$$

Let us now define the elements of these matrices.

$\boldsymbol{D^0}$**.** We first observe that a change of the state of a connection coincides always with the arrival of a cell for that connection. A connection that becomes active starts sending a packet immediately. A connection becoming silent is just finishing sending a packet, i.e., the last cell of the packet arrives in the CLS. This implies that the state of the arrival process will not change if no cell arrives, i.e.,

$$d_{ij}^{0} = 0 \qquad (i \neq j, 0 \leq i, j \leq m) . \tag{6.11}$$

If i connections are active, the probability that one of them sends a cell equals i/m. Furthermore, we define the probability that one of the silent connections becomes active and hence sends a cell as δ_i. Thus we can state for the probability that no cell arrives while i connections are active:

$$d_{ii}^{0} = 1 - \frac{i}{m} - \delta_i \qquad (0 \leq i \leq m) . \tag{6.12}$$

$\boldsymbol{D^{al}}$**.** From Eq. 6.2, it can be seen that a connection becomes silent with probability is. This can only occur if a last cell from one of the active connections arrives, i.e.,

$$d_{ij}^{al} = is \qquad (0 \leq j = i-1 \leq m-1) . \tag{6.13}$$

The probability of a cell arrival from one of the active connections is i/m. Recalling that the probability of a cell being the last one of a packet has been defined as p, and taking Eq. 6.13 into account, we define

$$d_{ii}^{al} = p\frac{i}{m} - is \qquad (0 \leq i \leq m) . \tag{6.14}$$

$\boldsymbol{D^{an}}$**.** Similarly, we can define for the probability of the arrival of a cell which is not the last one of a packet:

$$d_{ii}^{an} = (1-p)\frac{i}{m} \qquad (0 \le i \le m) \ . \tag{6.15}$$

$\boldsymbol{D^{sl}}$**.** As stated before, the probability that a cell from one of the silent connections arrives is defined as δ_i. With probability ms the new connection becomes immediately silent again, i.e.,

$$d_{ii}^{sl} = ms\delta_i \qquad (0 \le i \le m) \ . \tag{6.16}$$

The probability that a silent connection becomes active because of the arrival of a cell which is also the last (and only) one of a packet is

$$d_{ij}^{sl} = (p-ms)\,\delta_i \qquad (1 \le j = i+1 \le m) \ . \tag{6.17}$$

$\boldsymbol{D^{sn}}$**.** Finally, the probability that a silent connection becomes active because of the arrival of a cell which is not the last one of a packet is

$$d_{ij}^{sn} = (1-p)\,\delta_i \qquad (1 \le j = i+1 \le m) \ . \tag{6.18}$$

All other elements of $\boldsymbol{D}^{al}$, $\boldsymbol{D}^{an}$, $\boldsymbol{D}^{sl}$, and $\boldsymbol{D}^{sn}$ are zero.

We can derive the probability of a connection becoming active, δ_i, by recognizing that it equals the probability of a connection becoming active and staying active plus the probability of a connection becoming active and becoming silent again. These probabilities have been given in Eq. 6.2 and Eq. 6.16 respectively, so that

$$\delta_i = (m-i)\,r + \delta_i ms \ . \tag{6.19}$$

Consequently, we can derive

$$\delta_i = \frac{(m-i)\,r}{1-ms} \qquad (0 \le i \le m) \ . \tag{6.20}$$

Now we have completely specified the arrival process. We have specified the transition probabilities between the states of the modulating DTMC, and the probabilities of the events that can occur at these transitions. Such an event is the arrival of a cell, which is or is not the last one of a packet.

The arrival process can be interpreted as follows. A certain time-slot is assigned randomly to one of the m incoming connections. If this connection is active at the beginning of the slot, it fills the slot with a cell. Furthermore, it can become silent

during the time-slot with probability ms. If the connection is silent at the beginning of the time-slot, it can become active during the time-slot, stay active, and fill the time-slot with a cell with probability mr. Hence, a silent connection has a probability r to become active and stay active in an arbitrary time-slot, because it can only become active when the slot has been assigned to it, which happens with probability $1/m$. Similarly, an active connection has a probability s to become silent in an arbitrary time-slot. Finally, an active connection has a probability $1/m$ to send a cell in an arbitrary time-slot.

Let $\underline{\pi}^D$ be the stationary state probability vector of the underlying DTMC of the arrival process, i.e.,

$$\underline{\pi}^D \boldsymbol{D} = \underline{\pi}^D \text{, and} \tag{6.21}$$

$$\underline{\pi}^D \bar{e} = 1 \text{,} \tag{6.22}$$

where $\bar{e}$ is a column vector of ones. The i-th element of $\underline{\pi}^D$, π_i^D, is similar to the stationary probability for the continuous-time M/M/∞//M queue (see [89]: Section 3.9):

$$\pi_i^D = \frac{\left(\frac{r}{s}\right)^i \binom{m}{i}}{(1 + r/s)^m} \text{.} \tag{6.23}$$

Finally, we define the fundamental cell arrival probability p_{arrival} as the probability that a cell arrives in an arbitrary slot. It is given by

$$p_{\text{arrival}} = \underline{\pi}^D \boldsymbol{D}^1 \bar{e} \text{.} \tag{6.24}$$

Eq. 6.24 can be interpreted as the sum of the probabilities of all transitions that correspond to a cell arrival. It can not be easily simplified, because of the terms for arrivals from silent connections that become active.

6.1.2 The System Model

In the previous section, we have defined the probabilities of making a certain transition in the modulating arrival process, and having a cell arrival of a certain type (a cell from one of the active, or one of the silent connections, which is, or is not the last one of a packet) or having no arrival. In this section we describe what happens to the reassembly buffer if such a transition occurs. We assume that the buffer can contain at most n cells.

The state of the buffer can exactly be defined by the number of cells that are stored for each of the m connections. For silent connections, this number will be 0, for active connections, this number corresponds to the number of cells received from the packet currently being reassembled. Indeed, we have modelled the system in this way. However, we have specified and analysed this so-called *overall model*, using Stochastic Activity Networks, and the software tool UltraSAN ([117]), which allows for a description at a higher level. The problem of the overall model is its state space, which has a size of order $O(m^n)$.

In an attempt to reduce the state space of our model, we lump all those states that have the same total number of cells in the buffer. By this lumping, we loose the Markovian property of our model, i.e., the probability of a certain transition is no longer totally independent of previous states. However, we approximate the model as if this property was still valid. In fact only few transitions in the model do depend on previous states the system was in. In Section 6.2, we will compare results from this so-called *lumped model* with results from the overall model, to investigate the accuracy of the approximation.

The lumped model is described by the DTMC $\{A_t, N_t\}$, where A_t is defined as in the previous section, and $\{N_t, t=0, 1, 2, \ldots\}$ gives the number of cells in the buffer at the beginning of time-slot t. Furthermore, $N_t = 0$ if $A_t = 0$, and $N_t \in \{0, 1, \ldots, n\}$ if $A_t > 0$. The number of states in the lumped model equals $m(n+1)+1$, i.e., it is of order $O(mn)$.

Let the states of this DTMC be ordered lexicographically ((0,0), (1,0), (1,1),..., (1,n),..., (m,0), (m,1),..., (m,n)). Then the transition matrix $\boldsymbol{T}$ of the model can be written as follows:

$$\boldsymbol{T} = \begin{bmatrix} t_{00} & t_{01} & \cdots & t_{0j} & \cdots & t_{0m} \\ t_{10} & \boldsymbol{T}_{11} & \cdots & \boldsymbol{T}_{1j} & \cdots & \boldsymbol{T}_{1m} \\ \vdots & \vdots & . & \vdots & . & \vdots \\ t_{i0} & \boldsymbol{T}_{i1} & \cdots & \boldsymbol{T}_{ij} & \cdots & \boldsymbol{T}_{im} \\ \vdots & \vdots & . & \vdots & . & \vdots \\ t_{m0} & \boldsymbol{T}_{m1} & \cdots & \boldsymbol{T}_{mj} & \cdots & \boldsymbol{T}_{mm} \end{bmatrix} . \tag{6.25}$$

Here, t_{00} is the probability that the buffer is empty at the end of a slot with all connections silent, given that the buffer was empty at the beginning of the slot and that all connections were silent. It can be expressed in the transition probabilities of the arrival process (Section 6.1.1) as follows:

$$t_{00} = d_{00}^{0} + d_{00}^{sl} \ . \tag{6.26}$$

t_{0j} is a row vector of which the l-th element $(0 \le l \le n)$ gives the probability that at the end of a slot l cells are in the buffer and j connections are active, given an empty buffer and no active connections at the beginning of the slot. This can only happen if a cell from one of the silent connections arrives. If the cell is the last one of a packet, the buffer remains empty. Otherwise the buffer will contain a single cell at the end of the slot, i.e.,

$$t_{0j} = d_{0j}^{sl}\,[1, 0, 0, \ldots, 0] + d_{0j}^{sn}\,[0, 1, 0, \ldots, 0] \ . \tag{6.27}$$

t_{i0} is a column vector of which the k-th element $(0 \le k \le n)$ defines the probability that at the end of a slot the buffer is empty and no connections are active, given k cells in the buffer and i active connections at the beginning of the slot. Since this can only happen upon arrival of a cell from one of the active connections which is the last one of a packet, the column vector can be defined as

$$t_{i0} = d_{i0}^{al}\bar{e} \ , \tag{6.28}$$

where $\bar{e}$ is again a column vector of ones.

$\boldsymbol{T}_{ij}$ is an $(n+1) \times (n+1)$ matrix of which the (k,l)-th element defines the probability that the buffer contains l cells and j connections are active at the end of a slot, given k cells in the buffer and i active connections at the beginning of the slot. We define this matrix by

$$\boldsymbol{T}_{ij} = d_{ij}^{0}\boldsymbol{B}_{i}^{0} + d_{ij}^{al}\boldsymbol{B}_{i}^{al} + d_{ij}^{an}\boldsymbol{B}_{i}^{an} + d_{ij}^{sl}\boldsymbol{B}_{i}^{sl} + d_{ij}^{sn}\boldsymbol{B}_{i}^{sn} \ , \tag{6.29}$$

where $\boldsymbol{B}_{i}^{0}$ is a probability matrix that describes the transition probabilities of the buffer contents (N_t), given that no cell arrives in a slot, and i connections are active at the beginning of the slot. Similarly $\boldsymbol{B}_{i}^{al}$, $\boldsymbol{B}_{i}^{an}$, $\boldsymbol{B}_{i}^{sl}$, and $\boldsymbol{B}_{i}^{sn}$ describe the transition probabilities of the buffer contents given the arrival of a cell of the corresponding type (see Table 6-2) and i active connections at the beginning of the slot. In conclusion, $\boldsymbol{T}_{ij}$ can be interpreted as the sum of a number of terms, where each term consists of two factors; the first one to express the probability of a certain event (e.g., the arrival of a cell of a certain type), and the second one to express the impact of that event on the buffer contents.

Let us now define the elements of the ***B***-matrices.

$\boldsymbol{B_i^0}$**.** Since the buffer contents do not change if no cell arrives, we define

$$\boldsymbol{B}_i^0 = \boldsymbol{I} \qquad (1 \le i \le m) \ , \tag{6.30}$$

where $\boldsymbol{I}$ is the $(n+1) \times (n+1)$ identity matrix.

$\boldsymbol{B_i^{sl}}$**.** Since a cell arriving from one of the silent connections is always the first one of a packet, the buffer contents does not change if it is also the last one (a single cell packet),

$$\boldsymbol{B}_i^{sl} = \boldsymbol{I} \qquad (1 \le i \le m) \ . \tag{6.31}$$

$\boldsymbol{B_i^{sn}}$**.** If the cell arriving from one of the silent connections is not the last one of a packet, it is added to the buffer, unless the buffer is already full, i.e.,

$$\boldsymbol{B}_i^{sn} = \begin{bmatrix} 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & . & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 \\ 0 & 0 & 0 & 0 & \dots & 1 \end{bmatrix} \qquad (1 \le i \le m) \ . \tag{6.32}$$

$\boldsymbol{B_i^{al}}$**.** For the arrival of a cell of one of the active connections, things are a bit more complicated. Let us first consider the case of only one active connection at the beginning of the slot. If a cell arrives, which is the last one of a packet, the buffer will be emptied, because all the cells in the buffer belonged to that single packet, i.e.,

$$\boldsymbol{B}_1^{al} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 1 & 0 & \dots & 0 \\ \vdots & \vdots & . & \vdots \\ 1 & 0 & \dots & 0 \end{bmatrix} . \tag{6.33}$$

If several connections are active, it is not known from the state of our model how many cells are removed from the buffer upon arrival of the last cell of a packet. All other cells belonging to that packet, which are already in the buffer, will be removed upon the arrival. However, we lost the information concerning the size of the packet by lumping all states which had the same total number of cells in the

buffer. In the sequel we will determine the probability distribution for this size, and consequently for the number of cells to be removed from the buffer.

Looking back at previous arrivals on the connection until a cell is encountered that is the last one of a packet, i.e., the last one of the previous packet, can be seen as a series of Bernoulli trials. Hence, for each of the i active connections, the number of received cells since (including) the last one of the previous packet has a geometric distribution with parameter p. So, assuming that no cells have been lost, for each active connection, the number of cells in the buffer plus one has this distribution. Consequently, the total number of cells in the buffer plus i has a Pascal distribution with parameters p and i, since the sum of a number of geometrically distributed random variables has a Pascal distribution (see [84]: Section 29.14). Let us define the random variable denoting the number of cells in the buffer that are to be removed because of the arrival of the last cell of a packet as X. Furthermore, let us define the random variable denoting the number of cells in the buffer originated by i connections as Y_i. Now we can state for the probability that l cells remain in the buffer after the arrival of a last cell from one of the active connections, given k cells in the buffer before the arrival and i active connections, i.e., for the (k,l)-th element of $\boldsymbol{B}_i^{al}$:

$$b_{ikl}^{al} = P\{X = k-l \,|\, Y_i = k\} = \frac{P\{X = k-l, Y_i = k\}}{P\{Y_i = k\}} \qquad (0 \le l \le k \le n) \ . \tag{6.34}$$

By conditioning on X, we obtain

$$b_{ikl}^{al} = \frac{P\{Y_i = k \,|\, X = k-l\}\, P\{X = k-l\}}{P\{Y_i = k\}} \ . \tag{6.35}$$

By recognizing that Y_i is the sum of X and the number of cells in the buffer generated by $i-1$ other connections, i.e., $Y_i = Y_{i-1} + X$, we can derive

$$b_{ikl}^{al} = \frac{P\{Y_i - X = l\}\, P\{X = k-l\}}{P\{Y_i = k\}} = \frac{P\{Y_{i-1} = l\}\, P\{X = k-l\}}{P\{Y_i = k\}} \ . \tag{6.36}$$

Let us now modify this expression by adding constants to both sides of the equalities:

$$b_{ikl}^{al} = \frac{P\{Y_{i-1} + (i-1) = l + (i-1)\}\, P\{X + 1 = k-l+1\}}{P\{Y_i + i = k + i\}} \ . \tag{6.37}$$

By recalling that $X+1$ has a geometric distribution with parameter p, $Y_i + i$ has a Pascal distribution with parameters p and i, and $Y_{i-1} + (i-1)$ has a Pascal distribution with parameters p and $i-1$, we can derive from Eq. 6.37:

$$b_{ikl}^{al} = \frac{\left(\binom{l+i-2}{i-2} p^{i-1}(1-p)^{l}\right)\left((1-p)^{k-l}p\right)}{\binom{k+i-1}{i-1} p^{i}(1-p)^{k}} = \frac{\binom{l+i-2}{i-2}}{\binom{k+i-1}{i-1}} . \qquad (6.38)$$

Of course, the number of cells in the buffer will never increase upon arrival of a cell which is the last one of a packet, i.e.,

$$b_{ikl}^{al} = 0 \qquad (0 \le k < l \le n) . \qquad (6.39)$$

Now we are able to define the matrix $\boldsymbol{B}_i^{al}$ in terms of its elements:

$$\boldsymbol{B}_i^{al} = \left[b_{ikl}^{al}\right]_{(n+1)\times(n+1)} \qquad (2 \le i \le m,\ 0 \le k, l \le n) . \qquad (6.40)$$

$\boldsymbol{B_i^{an}}$. For the specification of $\boldsymbol{B}_i^{an}$, we need to observe that an arriving cell is stored in the buffer if it is not the last one of a packet, and the buffer is not already full. If the arriving cell finds the buffer full, all cells belonging to the same packet are removed from the buffer. Because of the geometric, and hence memoryless, distribution of the packet length, this implies that the distribution of the number of cells to be removed from the buffer is in this case as if the arriving cell was the last one of a packet, i.e.,

$$\boldsymbol{B}_1^{an} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & . & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 1 & 0 & 0 & \dots & 0 \end{bmatrix} , \text{ and} \qquad (6.41)$$

$$\boldsymbol{B}_i^{an} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & . & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ b_{in0}^{al} & b_{in1}^{al} & b_{an2}^{al} & \dots & b_{inn}^{al} \end{bmatrix} \qquad (2 \le i \le m) . \qquad (6.42)$$

This concludes the specification of the DTMC that describes the lumped model. Let us now continue with the analysis of the model.

Let $\underline{\pi}^T$ be the stationary state probability vector of the DTMC. It can be found by solving for

$$\underline{\pi}^T \boldsymbol{T} = \underline{\pi}^T \,, \tag{6.43}$$

and

$$\underline{\pi}^T \bar{e} = 1 \,. \tag{6.44}$$

$\underline{\pi}^T$ has the following form:

$$\underline{\pi}^T = \left[\pi_0^T, \underline{\pi}_1^T, \ldots, \underline{\pi}_i^T, \ldots, \underline{\pi}_m^T\right] \,, \tag{6.45}$$

where π_0^T is a scalar, denoting the probability that all connections are silent, and hence the buffer is empty. $\underline{\pi}_i^T$ is a vector of length $n + 1$, of which the k-th element (π_{ik}^T) denotes the probability that i connections are active and the buffer contains k cells.

Having obtained the stationary state probability vector of the model, we are able to derive performance measures from our model. The distribution of the buffer occupation can easily be obtained. This distribution ($\underline{\pi}^B$) can be defined as follows:

$$\underline{\pi}^B = \left[\pi_0^B, \pi_1^B, \ldots, \pi_k^B, \ldots, \pi_n^B\right] \text{, with} \tag{6.46}$$

$$\pi_0^B = \pi_0^T + \sum_{i=1}^{m} \pi_{i0}^T \text{, and} \tag{6.47}$$

$$\pi_k^B = \sum_{i=1}^{m} \pi_{ik}^T \qquad (1 \le k \le n) \,. \tag{6.48}$$

The measure, we aim at in the modelling and analysis of the reassembly buffer is the packet loss probability. It allows us to dimension the reassembly buffer in a proper way. Packet loss occurs when an arriving cell of the packet cannot be stored any more because the buffer is full. The other cells of the packet in the buffer are then also removed.

If a cell which is not the last one of a packet, arrives when the buffer is full, a batch of cells is removed from the buffer, i.e., is lost. In our model, after the loss of a batch of cells, which will actually result in packet loss, new cells of that packet

will still be stored in the buffer until the last cell of that packet arrives. This implies that another batch of cells of the same packet can be lost.

Let the random variable L denote the number of lost batches for an arbitrary arriving packet. First, we derive the expected value of L, given i active connections. We do this by conditioning on the packet length:

$$E[L|A_t = i] = \sum_{j=1}^{\infty} E[L|\text{packet has length } j \wedge A_t = i]\, P\{\text{packet has length } j|A_t = i\} \quad (6.49)$$

We assume that each arriving cell has the same probability of causing a batch loss. Furthermore, the packet length is independent of the number of active connections. Consequently, we can derive

$$E[L|A_t = i] = \sum_{j=1}^{\infty} j\, P\{\text{batch loss upon cell arrival}|A_t = i\}\, P\{\text{packet has length } j\}$$

$$= P\{\text{batch loss upon cell arrival}|A_t = i\} \sum_{j=1}^{\infty} j P\{\text{packet has length } j\} \quad . \quad (6.50)$$

The last factor of Eq. 6.50 equals the mean packet length, i.e.,

$$E[L|A_t = i] = \frac{P\{\text{batch loss upon cell arrival}|A_t = i\}}{p} \quad . \quad (6.51)$$

As said above, a batch is lost if a cell which is not the last one of a packet arrives when the buffer is full. Hence, Eq. 6.51 can be rewritten as

$$E[L|A_t = i] = \frac{P\{(\text{cell finds buffer full} \wedge \text{cell is not the last one of a packet})|A_t = i\}}{p}$$

$$= \frac{P\{\text{cell finds buffer full}|A_t = i\}\, P\{\text{cell is not the last one of a packet}|A_t = i\}}{p} \quad . \quad (6.52)$$

Recall that the probability of a cell being the last one of a packet equals p, and is independent of the number of active connections. This implies that Eq. 6.52 can be rewritten as

$$E[L|A_t = i] = \frac{1-p}{p} P\{\text{cell finds buffer full}|A_t = i\} \quad . \quad (6.53)$$

If no connections become active or silent, cells arrive according to a Bernoulli process. It is known that Bernoulli arrivals see time averages (BASTA), so that Eq. 6.53 can be written as

$$E[L|A_t = i] = \frac{1-p}{p} P\{\text{buffer full}|A_t = i\} = \frac{1-p}{p} \frac{P\{\text{buffer full} \wedge A_t = i\}}{P\{A_t = i\}} . \quad (6.54)$$

Let $\underline{p}_{\text{full}}$ be a vector of which the i-th element $(0 \le i \le m)$ is the probability that the buffer is full, and i connections are active. $\underline{p}_{\text{full}}$ can be defined as follows:

$$\left(\underline{p}_{\text{full}}\right)_0 = 0 \text{ , and} \quad (6.55)$$

$$\left(\underline{p}_{\text{full}}\right)_i = \pi_{in}^T \qquad (1 \le i \le m) . \quad (6.56)$$

Substituting $\underline{p}_{\text{full}}$ in Eq. 6.54 yields

$$E[L|A_t = i] = \frac{1-p}{p} \frac{\left(\underline{p}_{\text{full}}\right)_i}{P\{A_t = i\}} . \quad (6.57)$$

Now we are able to obtain $E[L]$, the expected number of lost batches for an arbitrary arriving packet. Since L is concerned with arriving packets, we only consider those slots in which packets arrive:

$$E[L] = \sum_{i=0}^{m} E[L|A_t = i] P\{A_t = i \,|\, \text{packet arrival}\} . \quad (6.58)$$

This can be rewritten as

$$E[L] = \sum_{i=0}^{m} E[L|A_t = i] \frac{P\{\text{packet arrival}|A_t = i\} P\{A_t = i\}}{P\{\text{packet arrival}\}} . \quad (6.59)$$

Every cell arrival is with probability p a packet arrival, regardless of the number of active connections, i.e.,

$$E[L] = \sum_{i=0}^{m} E[L|A_t = i] \frac{pP\{\text{cell arrival}|A_t = i\} P\{A_t = i\}}{p\, p_{\text{arrival}}} . \quad (6.60)$$

Substituting Eq. 6.57, and recognizing that the arrival probability, given i active connections, is by definition given by the sum of all elements on the i-th row of the matrix $\boldsymbol{D}^1$, yields

$$E[L] = \sum_{i=0}^{m} \frac{1-p}{p} \left(\underline{p}_{\text{full}} \right)_i \frac{P\{\text{cell arrival} | A_t = i\}}{p_{\text{arrival}}} = \frac{1-p}{p} \frac{\underline{p}_{\text{full}} \boldsymbol{D}^1 \bar{e}}{p_{\text{arrival}}} \quad . \tag{6.61}$$

Packet loss occurs if at least one batch of cells is lost for a packet, i.e., if $L \geq 1$. The probability that an arbitrary packet is lost ($p_{\text{packet loss}}$), can be obtained by conditioning:

$$\begin{aligned} E[L] &= E[L|L=0] P\{L=0\} + E[L|L \geq 1] P\{L \geq 1\} \\ &= E[L|L \geq 1] P\{L \geq 1\} \quad , \end{aligned} \tag{6.62}$$

so that

$$p_{\text{packet loss}} = P\{L \geq 1\} = \frac{E[L]}{E[L|L \geq 1]} \quad . \tag{6.63}$$

Recall that we assume that all arriving cells have an equal probability of causing the loss of a batch of cells. Hence, because of the memoryless property of the packet length distribution, a packet that has already lost a batch remains to have the same probability of losing another batch of cells, i.e.,

$$E[L|L \geq 1] = 1 + E[L] \quad . \tag{6.64}$$

Substituting Eq. 6.64, and Eq. 6.61 in Eq. 6.63 yields the packet loss probability:

$$p_{\text{packet loss}} = \frac{(1-p)\, \underline{p}_{\text{full}} D^1 \bar{e}}{p\, p_{\text{arrival}} + (1-p)\, \underline{p}_{\text{full}} D^1 \bar{e}} \quad . \tag{6.65}$$

6.1.3 Model Decomposition

The model described in Section 6.1.2 has a state space of $m(n+1)+1$ states. A large computational effort is required to solve its balance equations (Eq. 6.43 and Eq. 6.44). In the sequel, we will show how the stationary state probability vector of this matrix can be approximated by solving m systems of equations of size $n+1$, and one of size $m+1$. We make use of a property of our model that is

called near-complete decomposability, and come up with a new model, which will be referred to as *decomposed model*.

If the values of r and s are small compared to $1/m$, the matrix $\boldsymbol{T}$ is nearly completely decomposable ([24]). This means that it can be expressed as

$$\boldsymbol{T} = \boldsymbol{T}^* + \varepsilon \boldsymbol{E}, \qquad \varepsilon \ll 1 \ , \tag{6.66}$$

where $\boldsymbol{T}^*$ is a stochastic matrix of block diagonal form, i.e.,

$$\boldsymbol{T}^* = \begin{bmatrix} t_0^* & \underline{0} & \dots & \underline{0} & \dots & \underline{0} \\ \bar{0} & \boldsymbol{T}_1^* & \dots & \mathbf{0} & \dots & \mathbf{0} \\ \vdots & \vdots & . & \vdots & . & \vdots \\ \bar{0} & \mathbf{0} & \dots & \boldsymbol{T}_i^* & \dots & \mathbf{0} \\ \vdots & \vdots & . & \vdots & . & \vdots \\ \bar{0} & \mathbf{0} & \dots & \mathbf{0} & \dots & \boldsymbol{T}_m^* \end{bmatrix} . \tag{6.67}$$

Furthermore, $\boldsymbol{E}$ is a matrix with zero row sums. The elements of $\boldsymbol{E}$ that fall within the diagonal blocks defined in the structure of $\boldsymbol{T}^*$ are all non-positive, the elements outside these blocks are non-negative. All elements of $\boldsymbol{E}$ have an absolute value less than or equal to one. We have thus approximated $\boldsymbol{T}$ by $m+1$ subsystems with transition matrix $\boldsymbol{T}_i^*$, of which the states can be interpreted as describing the number of cells in the buffer given i active connections. Because of the definition of $\boldsymbol{T}$, the probabilities of transition between subsystems can be obtained from the transition matrix of the modulating arrival process ($\boldsymbol{D}$).

Let $\underline{\pi}_i^*$ be the stationary state probability vector of a DTMC with transition matrix $\boldsymbol{T}_i^*$, i.e.,

$$\underline{\pi}_i^* \boldsymbol{T}_i^* = \underline{\pi}_i^* \ , \text{and} \tag{6.68}$$

$$\underline{\pi}_i^* \bar{e} = 1 \ . \tag{6.69}$$

The stationary probability vector of our lumped model ($\underline{\pi}^T$) can then be approximated by

$$\underline{\pi}_i^T \approx \pi_i^D \underline{\pi}_i^* \qquad (1 \le i \le m) \ , \text{and} \tag{6.70}$$

$$\pi_0^T \approx \pi_0^D \ , \tag{6.71}$$

i.e., the probability of having i active connections and k cells in the buffer can be approximated by the product of the probability of i active connections and the approximation of the probability of k cells in the buffer given i active connections. The error of this approximation is known to go to zero with ε ([24]).

Let us now define the constituting matrices of $\boldsymbol{T}^*$. As stated above, $\boldsymbol{T}_i^*$ can be interpreted as the transition matrix of a DTMC of which the states are describing the number of cells in the buffer, given i active connections. In this perspective, it seems to be an appropriate choice to define $\boldsymbol{T}_i^*$ in the same way as $\boldsymbol{T}_{ii}$ (Eq. 6.29), taking into account the fact that no connections become active or silent. Hence, compared to Eq. 6.29, the last two terms should be omitted, because they correspond to a transition where a connection becomes silent. Similarly, the coefficients of the $\boldsymbol{B}$-matrices in Eq. 6.29 should be slightly adjusted. We define $\boldsymbol{T}_i^*$ as follows for $1 \leq i \leq m$:

$$\boldsymbol{T}_i^* = \left(1 - \frac{i}{m}\right)\boldsymbol{B}_i^0 + \frac{i}{m}p\boldsymbol{B}_i^{al} + \frac{i}{m}(1-p)\,\boldsymbol{B}_i^{an} \ . \tag{6.72}$$

This can be interpreted as follows. $\boldsymbol{T}_i^*$ is a matrix of which the (k, l) -th element specifies the probability that the buffer contains l cells at the end of a slot, given that it contained k cells at the beginning of the slot, and given that i connections are active, that no silent connections become active, and that no active connections become silent. With probability i/m a cell arrives, which is the last one of a packet with probability p, and which is not with probability $1-p$.

For the border case of $i = 0$, we have to define a 1×1 matrix, of which the only element should be one in order to have a stochastic matrix, i.e.,

$$t_0^* = 1 \ . \tag{6.73}$$

So, if no connections are active, and no silent connections become active, the buffer will remain empty with probability 1.

We have decomposed the lumped model according to the time scale at which transitions take place. The modulation of the arrival process is relatively slow, because of the small values that are expected for r and s. The rates at which the buffer contents change on the other hand are relatively high.

Let us now determine a value of ε, for which Eq. 6.66 holds. By realizing that all elements of the $\boldsymbol{B}$-matrices are smaller than or equal to 1, and that the coefficients in Eq. 6.29 sum to d_{ij}, it can be easily seen that

$$\mathrm{M}(\boldsymbol{T}_{ij}) \leq d_{ij} \ , \tag{6.74}$$

where $\mathrm{M}(A)$ denotes the maximum of the absolute values of the matrix elements, i.e., $\mathrm{M}(A) = \max_{i,j}(|a_{ij}|)$. Similarly, it can be seen that

$$\mathrm{M}\left(\boldsymbol{T}_i^* - \boldsymbol{T}_{ii}\right) \leq 1 - d_{ii} \ , \tag{6.75}$$

since the coefficient in Eq. 6.72 sum to 1. Hence, the maximum difference between the elements of $\boldsymbol{T}^*$ and $\boldsymbol{T}$ can be expressed as follows:

$$\mathrm{M}(\boldsymbol{T}^* - \boldsymbol{T}) \leq \mathrm{M}(\boldsymbol{D} - \boldsymbol{I}) \ . \tag{6.76}$$

It can easily be seen from Eq. 6.2 that the element of $\boldsymbol{D} - \boldsymbol{I}$ with the largest absolute value is either $m \times r$ or $m \times s$. Therefore, if

$$\varepsilon = m \times \max(r, s) \ , \tag{6.77}$$

a matrix $\boldsymbol{E}$ can be found such that Eq. 6.66 is valid. So, Eq. 6.71 is a good approximation if $1/r$ and $1/s$ are some orders of magnitude larger than m, i.e., the probability that a connection becomes active or silent in a time slot is sufficiently low. Indeed, this was a starting point of the analysis.

We finally have to adapt the definition of the packet loss probability slightly, so that it matches the decomposed model. Using the decomposition, Eq. 6.55 and 6.56 change to

$$\left(\underline{p}_{\text{full}}^*\right)_i = \pi_i^D \pi_{in}^* \ , \tag{6.78}$$

where π_{0n}^* is assumed to be zero. This is the probability that the buffer is full, given that the arrival process is in state i, times the probability that the arrival process is in state i. Furthermore, the fundamental cell arrival probability (p_{arrival}) changes slightly from its definition in Eq. 6.24, to

$$p_{\text{arrival}}^* = \underline{\pi}^D \bar{p}_a^* \ , \text{ where} \tag{6.79}$$

$$\bar{p}_a^* = \begin{bmatrix} 0 \\ 1/m \\ \vdots \\ i/m \\ \vdots \\ 1 \end{bmatrix} \ , \tag{6.80}$$

i.e., its *i*-th element gives the cell arrival probability in case of i active connections (see Eq. 6.72). Similarly, the factor in Eq. 6.61 representing the cell arrival probability given *i* active connections can be replaced by $\bar{p}_a{}^*$, i.e.,

$$E[L]^* = \frac{1-p}{p} \frac{\underline{p}_{\text{full}}{}^* \bar{p}_a{}^*}{p_{\text{arrival}}{}^*} \quad . \tag{6.81}$$

Taking Eq. 6.78, Eq. 6.79, and Eq. 6.81 into account, the packet loss probability (Eq. 6.65) changes to

$$p_{\text{packet loss}}{}^* = \frac{(1-p)\,\underline{p}_{\text{full}}{}^* \bar{p}_a{}^*}{p\,p_{\text{arrival}}{}^* + (1-p)\,\underline{p}_{\text{full}}{}^* \bar{p}_a{}^*} = \frac{(1-p)\,\underline{p}_{\text{full}}{}^* \bar{p}_a{}^*}{\left(p\underline{\pi}^D + (1-p)\,\underline{p}_{\text{full}}{}^*\right)\bar{p}_a{}^*} \quad . \tag{6.82}$$

Using Eq. 6.72, we can solve for $\underline{\pi}_i{}^*$. However, the expression depends on m, the total number of incoming connections. If we were able to remove m from the expression, we could use the result of Eq. 6.68 for different experiments, with different values of m. We replace the transition matrix $\boldsymbol{T}_i{}^*$ by a transition matrix $\boldsymbol{T}_i{}^{**}$, which specifies the transition probabilities at cell arrival instants only,

$$\boldsymbol{T}_i{}^{**} = p\boldsymbol{B}_i^{al} + (1-p)\,\boldsymbol{B}_i^{an} \, . \tag{6.83}$$

The DTMC with transition matrix $\boldsymbol{T}_i{}^{**}$ has the same stationary state probability vector as the one with $\boldsymbol{T}_i{}^*$, using Eq. 6.30 and Eq. 6.68:

$$\underline{\pi}_i{}^* \boldsymbol{T}_i{}^{**} = \underline{\pi}_i{}^* \left(\frac{m}{i} \left(\boldsymbol{T}_i{}^* - \left(1 - \frac{i}{m}\right) \boldsymbol{B}_i^0 \right) \right) = \frac{m}{i} \underline{\pi}_i{}^* \boldsymbol{T}_i{}^* - \frac{m-i}{i} \underline{\pi}_i{}^* \boldsymbol{I}$$

$$= \frac{m}{i} \underline{\pi}_i{}^* - \frac{m-i}{i} \underline{\pi}_i{}^* = \underline{\pi}_i{}^* \, . \tag{6.84}$$

6.2 Evaluation

Let us now present some of the results, obtained with the models described in the previous section. First, in Section 6.2.1, we evaluate the presented models with respect to their accuracy, and their computational complexity. In Section 6.2.2, we evaluate the loss probability of a reassembly buffer.

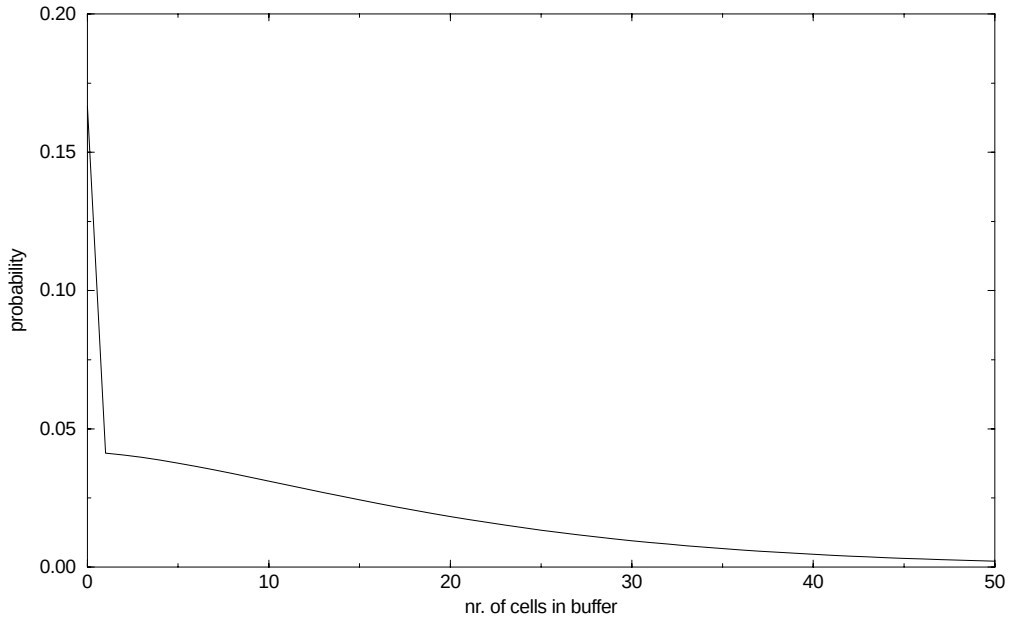


Figure 6-1: Buffer Occupancy Distribution (m=3, n=50)

6.2.1 Accuracy and Model Complexity

In order to get insight in the accuracy of the lumped model and the decomposed model, we compare results from these models with results from the overall model. The evaluation of the lumped and the decomposed model, based on the expressions given in Section 6.1, is done using two programs in the computing environment MATLAB ([97]). The evaluation of the overall model is performed using the software tool UltraSAN ([117]). Since the state space of the overall model grows very fast, we are only able to make comparisons for small m and n. The error bound for the decomposed model, which we have found in Eq. 6.77, increases linear in m. According to Eq. 6.77, the error is acceptable if $1/m$ is sufficiently larger than r and s. This is a realistic assumption, since the burst-silence behaviour of the connection is at another time-scale than the cell arrival behaviour.

To illustrate the accuracy of our simplified model, we give the buffer occupancy distribution for $m = 3$ and $n = 50$ (Figure 6-1). For the other parameters we have taken the values $r = s = 0.0001$ and $p = 0.1$, i.e., a mean packet length of 10 cells. Figure 6-2 shows the relative error in our approximation. It can be seen from the figure that the relative error is always below 0.1%.

It has been stated already in Section 6.1 that the overall model has a state space of order $O(m^n)$, and the lumped model of order $O(mn)$. The decomposed model

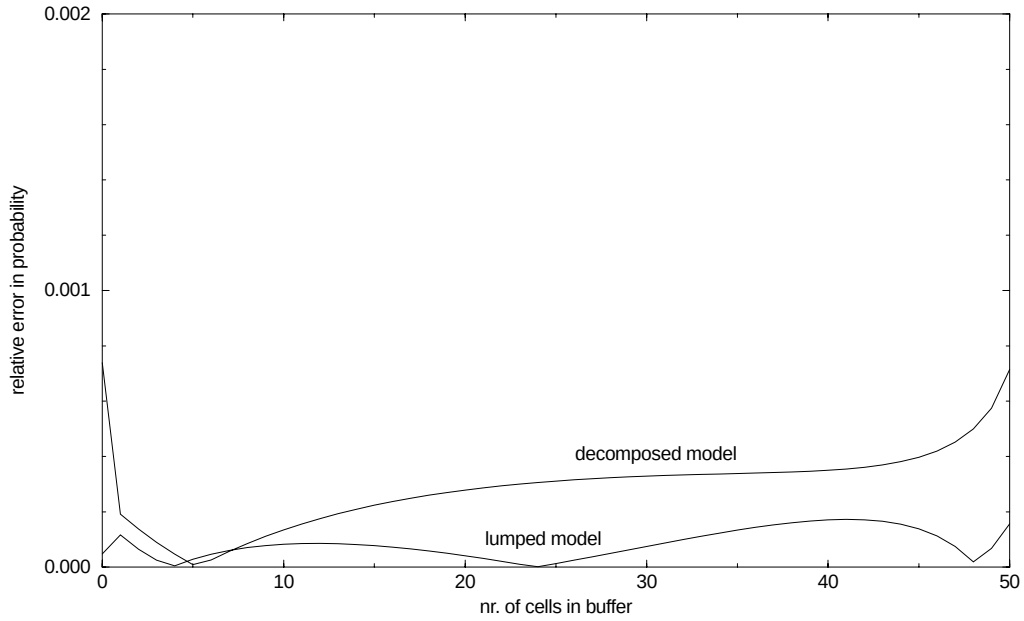


Figure 6-2: Relative Error in Buffer Occupancy Distribution compared to Overall Model

can be considered as consisting of m DTMCs of order $O(n)$. It is known that the steady-state probability vector of a DTMC with n states can be obtained with a computational complexity of order $O(n^2)$. Table 6-3 gives the order of complexity of the various models. It can easily be seen that with the overall model, it is only possible to obtain results for small m and n. The difference between the computational complexity of the lumped and the decomposed model is of the order of the number of incoming connections (m).

6.2.2 Packet Loss Probability

Question 4 in Section 5.1.2 was concerned with the required size of the reassembly buffer. Figure 6-3 shows the relation between the buffer length n, and the packet loss probability $p_{\text{packet loss}}$ for different values for the number of connections terminated in the CLSM (m). The values of r, s, and p are as in Section 6.2.1. The decomposed model is used to obtain the results of this figure.

It can be seen that for small m the logarithm of the loss probability is almost inversely proportional to the buffer length. Furthermore, the packet loss probability depends very much on m. The required buffer length for a certain loss probability is less than proportional to the number of incoming connections, e.g., in order to obtain a loss probability of 10^{-5}, the required buffer size is approximately 220 for $m = 10$, and 450 for $m = 40$. This can be explained by the fact that

model	order of state space size	order of computational complexity
overall model	$O(m^n)$	$O(m^{2n})$
lumped model	$O(mn)$	$O(m^2n^2)$
decomposed model	$m \times O(n)$	$O(mn^2)$

Table 6-3: Order of Complexity of the Overall, Lumped, and Decomposed Model

packets share buffer space, i.e., the maximum need for buffer space of the different connections does not occur simultaneously most of the time.

It can be concluded that the required buffer size for a given loss probability depends very much on the number of connections terminated in the CLSM. As a consequence, the loss probability also depends very much on the chosen implementation architecture. For implementation architectures 2, 4, and the new architecture 7, the required buffer size is significantly smaller than for the other architectures, because they have a smaller number of incoming connections per CLSM (m).

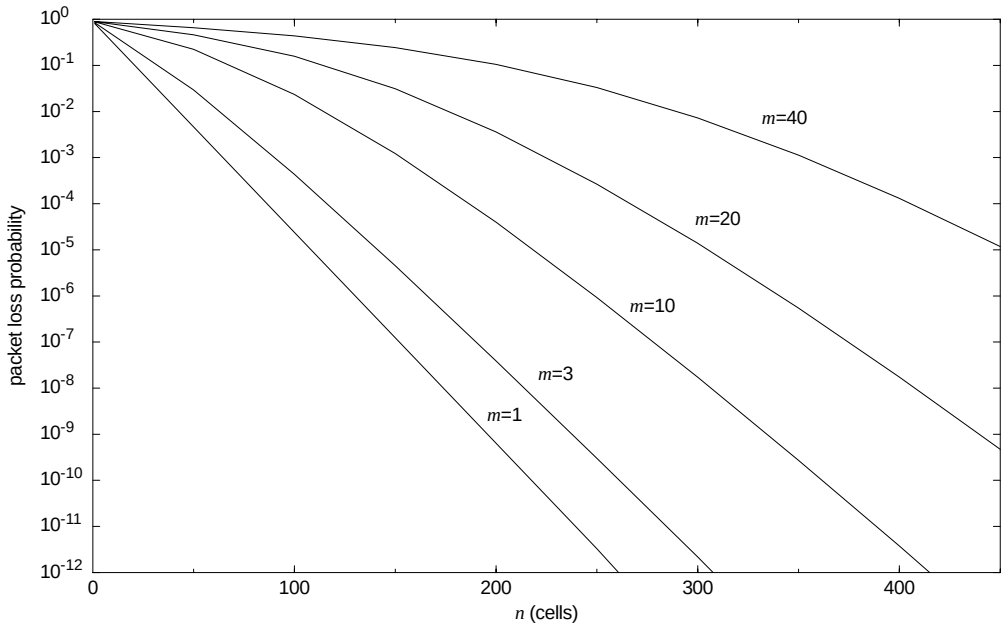


Figure 6-3: Packet Loss Probability

6.3 Summary and Concluding Remarks

In this chapter, we have modelled and analysed the reassembly buffer, which is used in a CLS operating in message mode to reassemble packets from the incoming cells. We have focused on the loss behaviour of the buffer. The overall model of this buffer has a very high computational complexity. We have presented two models with a much lower complexity, by applying lumping and decomposition techniques. The accuracy of the lumped and decomposed model have been evaluated. For typical values of the parameters, the relative error in the buffer occupancy distribution of both models is below 0.1%.

The developed models have been used to get insight in the loss probability of the buffer, given the number of incoming connections, the packet length, and the characteristics of the arrival process. For the used parameters, the analysis has shown that the required buffer space depends heavily on the number of connections terminated in a CLSM.

No extensive evaluations have been performed in this chapter. Here, the main purpose was to provide a tool, which can be used by a system designer to dimension the reassembly buffer. With a reasonable computational effort, the tool provides packet loss probabilities for a wide range of buffer sizes, given a certain load.

Today is the greatest day I've ever known.
Can't live for tomorrow, tomorrow's much too long.

Smashing Pumpkins - Siamese Dream

Chapter 7
Connection Management

ATM is used as the underlying technique to provide a broadband connectionless service to the users. However, ATM is a connection-oriented technique. Before the transfer of user data can take place, a connection needs to be established, and bandwidth needs to be reserved on this connection (Section 3.2.4). This is a function of the signalling and control system of the telecommunication network ([49], [101], [102]). The signalling system must be requested by entities in the BCL Network Layer to initiate the establishment of a connection, and to reserve bandwidth. This function will be referred to as connection management, and will be discussed in this chapter. We propose a candidate mechanism for the connection management function. Further, we model, analyse, and evaluate the mechanism with respect to its performance.

First, in Section 7.1, we will define the connection management function, and present the mechanism that can be used to perform the function. In Section 7.2 we model and analyse the mechanism. In Section 7.3, we evaluate the performance of the mechanism, and compare it to the performance of some conventional strategies. Finally, in Section 7.4, we give some concluding remarks.

7.1 General

In order to avoid confusion concerning the meaning of the term connection management, we will spend a separate subsection on explaining what the function comprises (Section 7.1.1). After that, in Section 7.1.2, we will propose a candidate mechanism for the function, and relate it to other, more conventional mechanisms. Finally, in Section 7.1.3, a number of performance measures will be identified, which will be used to evaluate the mechanisms.

7.1.1 The Connection Management Function

Connection Management is a function in a BCL Network Layer protocol entity that interacts with the signalling system of the ATM network to ensure that an AAL

connection to the proper destination, and with sufficient bandwidth, is available for the transfer of a BCL-PDU. The connection management function only interacts with the signalling system locally. It can ask the signalling system to establish a connection to a certain destination, to modify the bandwidth assigned to a connection, or to release a connection. In case of an establishment or a request for additional bandwidth, negotiation is done between the requesting protocol entity, the destination protocol entity, and the signalling system (on behalf of the ATM network). The signalling system confirms the success or failure of the request to the connection management function. No negotiation is needed for the release of a connection.

The actual establishment, maintenance, and release of AAL connections is performed by the signalling system. For this purpose, the signalling system must interact with AAL entities, and with ATM Layer entities in all systems along the route of the connection. In the ATM Layer, the AAL connection is supported by an ATM connection.

In case the indirect method of providing a connectionless service is employed (see Section 3.3.3), the AAL connections are established between BCL Network Layer entities in end-systems. In case of the direct method (see Section 3.3.4) connections are established between entities in end-systems and entities in CLSs, or between entities in CLSs mutually. All these connections together form a logical overlay network over the ATM network. At the level of this overlay network, ATM switches are no longer visible, only systems that implement the BCL Network Layer, and hence can terminate AAL connections are visible. Figure 7-1 shows an example of an overlay network. A line in the figure represents a set of two connections, one for each direction. Here, the indirect method is used between end-systems A and B. End-system B also employs the direct method, and therefore has a connection with CLS B. End-systems C and D are connected to CLS A, and both CLSs are also connected.

In order to maintain the overlay network of connections properly, the connection management function must analyse the need for the transfer of BCL-PDUs. Here a problem arises: in principle, the protocol entity does not have advance knowledge about the destination and the required bandwidth of the AAL connections it will need. This is because of the connectionless nature of the communication, where only individual BCL-PDUs are transferred, which are routed independently through the network.

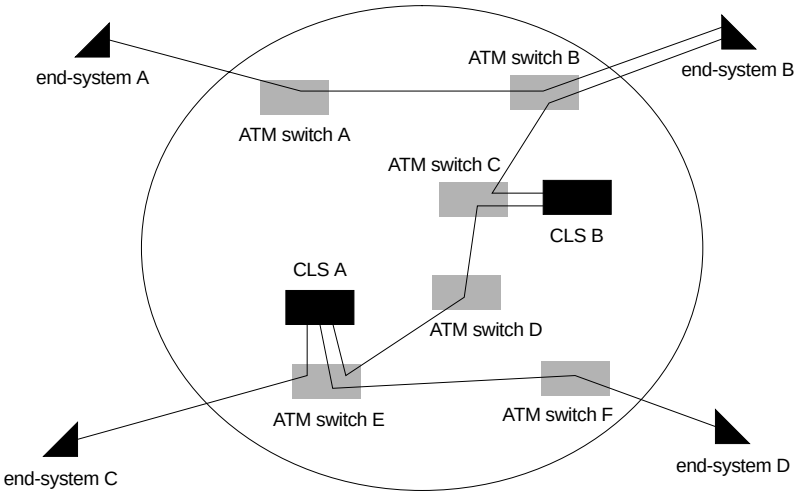


Figure 7-1: Example of a Connectionless Overlay Network

In practice, a certain correlation, in the destination as well as in the arrival time, can be expected due to the behaviour of (application) protocols in the higher layers. An application for which a single BCL-PDU is transferred, is likely to generate more data, so that more packets to the same destination will follow within a limited period of time. This expected correlation can be exploited by the connection management function, i.e., it can be used in requesting the establishment, modification, and release of AAL connections. For example, the expectation that there is a large amount of traffic between two LANs which have to be interconnected, can lead to the establishment of a permanent connection between these LANs. The new mechanism we present in the following section, exploits this correlation to a large extent.

7.1.2 Candidate Connection Management Mechanisms

Several mechanisms for setting up AAL connections for the transfer of BCL-PDUs can be envisaged[1]. The objective of such a mechanism is to minimize the load that is put on the ATM network for maintaining AAL connections. At the ATM level, ATM connections are used to support the AAL connections. Bandwidth that is reserved for these connections cannot be used for other (connection-oriented) applications. Furthermore, the load on the signalling system for establishing, modifying, and releasing connections must not be too high. Too many signalling operations could result in an overloaded signalling system, and hence in a degraded signalling performance, e.g., in a high connection setup delay. Finally,

[1] From now on we will use the term packet again for BCL-PDU.

the delay, experienced by packets in a BCL Network Layer entity, must be acceptable.

Let us now present the candidate connection management mechanisms. First, we describe two conventional mechanisms. After that, we introduce a novel one.

Connection per Packet (CpP)

A connection, necessary to transfer a packet, can be established as soon as the packet arrives at a protocol entity. The connection is released again immediately after the transfer of the packet. This mechanism does not exploit the expected correlation between packets. All necessary knowledge, i.e., destination and amount of data to be transported, is known at the moment the connection is established. Note that a packet will in general be sent in several ATM-SDUs (cells). A practical extension is to transfer subsequent packets with the same destination, arriving during the transmission time of the current packet, over the same connection.

Permanent Connection (PC)

Alternatively to the previous mechanism, a protocol entity can maintain one or more permanent connections to various possible destination entities. A packet arriving in this protocol entity is transferred on one of these connections. Note that the entity must maintain connections to all entities to which it must be able to transfer packets directly. Other entities may be reached in several steps, via intermediate entities.

The exact specification of the required bandwidth is a problem for this mechanism. In principle, the CL protocol entity has no advance knowledge about the arrival times and the lengths of the packets. It can only use information about subscription, and statistics to predict the required bandwidth. Optionally, the connection management mechanism may modify the bandwidth of the connection during the lifetime of a connection ([51]).

On-demand Connection with Delayed Release (OCDR)

In this mechanism an AAL connection will be established if a packet has arrived, and no connection to the proper destination protocol entity is available. The connection will not be released immediately after transferring the packet. It can be used for consecutive packets to the same destination as well. The connection

will be released if it has not been used for a certain period of time, the *holding time*.

This mechanism tries to exploit the expected correlation between the interarrival times of consecutive packets for the same destination entity. It assumes that the expected time until the next arrival is longer after the holding time has expired than immediately after a departure. Thus it can reduce the time a connection has to be maintained, compared to the PC mechanism, and at the same time, reduce the mean delay experienced by packets, compared to the CpP mechanism.

The CpP and PC mechanisms are special cases of this one. The OCDR mechanism with holding time zero is equivalent to the CpP mechanism with the mentioned extension. The PC mechanism is equivalent to the OCDR mechanism with an infinite holding time.

Discussion

The described mechanisms can be applied between two end-systems, as well as between an end-system and a CLS, and between two CLSs. The choice for a certain mechanism and the accompanying parameters (e.g., holding time) can be made for each pair of (source and destination) protocol entities[2] individually. The choice will depend on the expected arrival times and packet lengths of the traffic from the source protocol entity to the destination protocol entity. These depend heavily on the expected applications. Furthermore, the arrival times depend also on the extent to which traffic from different applications or end-systems can be multiplexed onto one connection. If the direct method of providing a connectionless service is used, packets to different end-systems can be multiplexed, e.g., on the connection to the first CLS, or between CLSs. For the indirect method, packets to different end-systems must use different connections, since end-to-end connections are used in this case. Note that an end-system can for instance be a router or bridge to a LAN, so that it does already multiplex the traffic of different LAN stations.

The amount of traffic that must be transferred over connections between CLSs in the connectionless overlay network can be expected to be so high that permanent connections must be maintained between these CLSs. Between which CLSs connections must be established, and how much bandwidth must be assigned to

[2] The terms source and destination do not necessarily refer to end-systems. They must be interpreted relative to the connection, and can as such refer to intermediate systems, i.e., CLSs.

the connections is a dimensioning problem that is similar to dimensioning problems that can be found in traditional data networks ([87], [88]).

In [51], it has been shown that it can be advantageous to modify the bandwidth assigned to the connections dynamically. In that article, we propose a mechanism for modifying the bandwidth, based on contents of the traffic shaping buffer, called Variable Bandwidth Connection (VBC) mechanism. Modelling and analysis of a simple version of the mechanism has revealed that it can reduce the average required bandwidth of a connection by 30 to 80%, without affecting the delay experienced by packets.

In this chapter, we focus on mechanisms for the establishment and release of fixed-bandwidth connections. The proposed OCDR mechanism is a mechanism that can provide for this. If the direct method of providing a connectionless service is employed, it can be used to connect end-systems to an Access CLS. It can be applied on the connection from end-system to CLS and from CLS to end-system independently. If the indirect method is employed, i.e., if end-systems need to be connected directly, and no CLSs are used, the OCDR mechanism can be used to control the connection between end-systems.

7.1.3 Performance Criteria

The three connection management mechanisms (CpP, PC, and OCDR) differ in a number of ways. In the following sections, we will investigate what the performance differences are. The following performance measures are important for the evaluation of the mechanisms.

- Average Delay
  The average delay is the mean time elapsing from the arrival of a packet in the traffic shaping buffer of a CLS until the departure of (the last cell of) the packet from the buffer. This delay consists of the time a packet spends in a buffer plus the time necessary for transmission of the consecutive cells of a packet on an outgoing connection. In Chapter 5, we have seen that the traffic shaping delay is one of the major components of the total delay experienced in a CLS.

- Average Reserved Bandwidth
  The average reserved bandwidth is the long time average of the bandwidth reserved on a connection between a source / destination pair of CLSs. Periods of time in which no connection is available are taken into account in this average, and are considered as periods during which the reserved bandwidth is zero.

The average reserved bandwidth will be strongly related to the costs of the service.

- Average Number of Connection Setups per Second
  The average number of connection setups per second is the long-time average of the number of times a connection is established per second. This measure is an indication for the load on the signalling system, which is also a cost factor for the provision of a connectionless service. Note that the number of requests for connection release, equals the number of requests for connection establishment.

7.2 Modelling and Analysis

In the previous section, we have presented OCDR as one of the candidate mechanisms for the connection management function. The purpose of this mechanism was to reduce the reserved bandwidth, compared to the PC mechanism, while reducing the delay, compared to the CpP mechanism. The purpose of this section is to investigate this effect quantitatively. Therefore, we model and analyse the mechanism to obtain results for the performance measures identified in Section 7.1.3. The modelling and analysis can also be used for the CpP and PC mechanisms, since these are special cases of the OCDR mechanism.

We focus on a single node. Furthermore, we only consider the packets that are destined for a single other node. In other words, we model the behaviour of the OCDR mechanism as far as a single pair of source and destination nodes is concerned. A connection between these entities may or may not exist. The connection is established if a packet arrives in the source node for the destination node under consideration. After finishing the transfer of a packet, the connection is released if no new packet for the destination arrives before the holding time expires.

We refrain from modelling the detailed behaviour at cell level, since the packet level behaviour is dominant at the time-scale that is relevant for the connection management mechanisms. Furthermore, the models in this section are continuous time models, since the slotted nature of ATM is not relevant at the considered time-scale.

Table 7-1 gives an overview of the model parameters for later reference. They will be introduced later on.

$1/\lambda$	average packet interarrival time
$1/\mu$	average packet transmission time
$1/r$	average connection holding time
$1/c$	average connection setup time
l	average packet length
$1/\alpha$	average burst time
$1/\beta$	average interburst time
n	number of Erlang stages for the holding time
m	number of Erlang stages for the connection setup time

Table 7-1: Model Parameters

In Section 7.2.1, we first discuss the workload of the model, i.e., the stream of packets arriving in the source node that are destined for the destination node. Two different types of workload are defined, Poisson traffic and bursty traffic. In Section 7.2.2, we present the modelling and analysis of the mechanism assuming the Poisson traffic. Next, in Section 7.2.3, we present the modelling and analysis assuming bursty traffic. Finally, in Section 7.2.4, we present an extension to the latter model, which is based on more realistic assumptions for the holding time of the OCDR mechanism and the time needed to establish a connection.

7.2.1 Workload

We adopt two different workload characterizations for the analysis of the OCDR mechanism. The first one, Poisson traffic, results in an analytically tractable performance model. It gives insight in the behaviour of the protocol. The second one, bursty traffic, gives a more realistic characterization of the expected traffic, and therefore more accurate performance measures.

Poisson Traffic

The consecutive interarrival times of packets are assumed to be independent, and exponentially distributed with mean $1/\lambda$. We are able to derive a closed-form solution for our model using this workload.

The operation of the OCDR mechanism is based on the assumption that there will be a correlation in arrival times of subsequently transferred packets. Poisson traffic does not have this property. Therefore, the advantages of the mechanism will not be revealed.

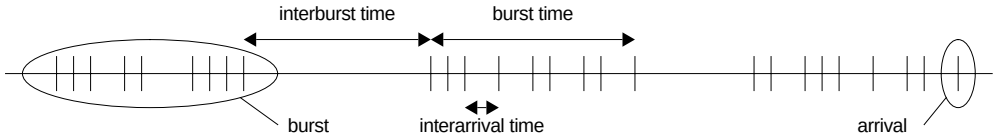


Figure 7-2: Behaviour of an Interrupted Poisson Process

Bursty Traffic

Here, we assume that packets arrive in bursts. In [83], it is shown that such a bursty traffic source, which is modelled by a "Train Model", provides a realistic description of the traffic on a local computer network (Ethernet). In [44], it is shown that a Markov Modulated Poisson Process (MMPP), which can also be used to describe the bursty traffic source is very well suited to represent correlations between subsequent arrivals. We prefer to model the traffic source with a simple type of MMPP, which is also referred to as an Interrupted Poisson Process (IPP). This arrival process approximates the "Train Model", and has the advantage of allowing for Markovian analysis.

Let the stochastic process $\{A(t), t \geq 0, A(t) \in \{0,1\}\}$ describe the arrival rate at time t. If $A(t) = 1$, the arrival process is said to be in a burst, and packets arrive at rate λ, i.e., with exponentially distributed interarrival times with mean $1/\lambda$. If $A(t) = 0$, the arrival process is said to be in an interburst period and no packets arrive. We assume the burst time and the interburst time to be exponentially distributed, with mean $1/\alpha$ and $1/\beta$ respectively (see Figure 7-2). It can easily be seen that the long-term mean of $A(t)$, $E[A]$ can be expressed as

$$E[A] = \frac{1/\alpha}{1/\beta + 1/\alpha} = \frac{\beta}{\alpha + \beta} . \qquad (7.1)$$

$E[A]$ can be interpreted as the fraction of time the arrival process is in the burst state. This implies that the mean arrival rate is given by $E[A]\lambda$.

Packet Length Distribution

We assume the packet length to be exponentially distributed. The mean packet length is l bits. Note that in a real situation, the packet length will have a discrete distribution. However, the exponential distribution can be seen as a continuous time equivalent of the geometric distribution.

7.2.2 OCDR under Poisson Traffic

Let us now present a model for the OCDR mechanism with Poisson arrivals. The concerned model is a Continuous Time Markov Chain (CTMC). We are able to derive a closed-form solution for the stationary state probabilities of this model.

When a packet arrives, and no connection exists, the connection management function invokes the signalling system in order to establish a connection. Setting up a connection is assumed to take an exponentially distributed time with mean $1/c$. When the establishment of the connection has been confirmed to the connection management function, the transmission of packets can be started. We assume that transmission of packets takes place at a rate of μl bits per second, i.e., every packet transmission takes an exponentially distributed time with mean $1/\mu$ seconds. After all packets have been sent, the connection will be released when the system is empty for an exponentially distributed holding time with mean $1/r$.

Let the stochastic process $\{N(t), t \geq 0, N(t) \in \aleph\}$ give the number of packets in the system at time t, i.e., the number of packet in the source entity, destined for the destination entity under consideration. Furthermore, let the stochastic process $\{V(t), t \geq 0, V(t) \in \{0, 1\}\}$ indicate whether or not a connection is available at time t. Then, the process $\{N(t), V(t)\}$ is a CTMC. In Figure 7-3 the resulting state transition diagram is depicted.

We are interested in the steady state behaviour of this CTMC. Let us denote the steady state probability distribution of this CTMC as $P(i, j)$:

$$P(i,j) = \lim_{t \to \infty} P\{N(t) = i \wedge V(t) = j\} \quad . \tag{7.2}$$

The following system of balance equations can be obtained (Eq. 7.3 through Eq. 7.6):

$$(c + \lambda) P(i, 0) = \lambda P(i-1, 0) \quad \text{, for } i \geq 1 \quad , \tag{7.3}$$

by equating the flow into state $(i, 0)$ to the flow out of the same state;

$$\mu P(i, 1) = \lambda (P(i-1, 1) + P(i-1, 0)) \quad \text{, for } i \geq 1 \quad , \tag{7.4}$$

by equating the flow across the boundary between the states with $N(t) \geq i$ and the states with $N(t) < i$;

$$rP(0, 1) = \lambda P(0, 0) \quad , \tag{7.5}$$

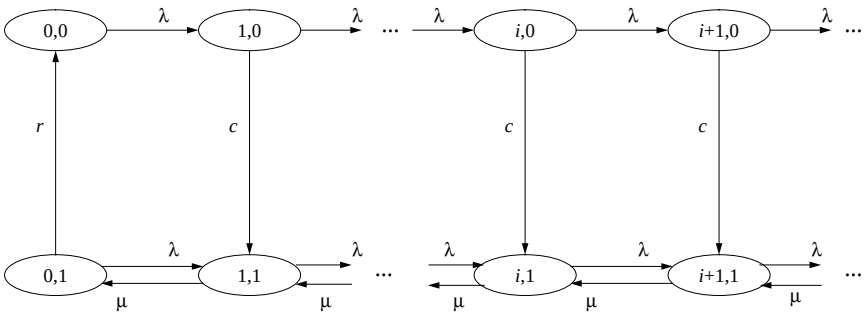


Figure 7-3: CTMC for OCDR under Poisson Traffic

by equating the flow into state $(0, 0)$ to the flow out of this state; and

$$\sum_{i=0}^{\infty} (P(i, 0) + P(i, 1)) = 1 , \tag{7.6}$$

for normalization.

We will relate all stationary probabilities to $P(0, 0)$, and use the following notation:

$$\rho = \frac{\lambda}{\mu} , \text{ and} \tag{7.7}$$

$$\sigma = \frac{\lambda}{\lambda + c} . \tag{7.8}$$

From Eq. 7.3 it follows directly that, for all *i*,

$$P(i, 0) = \sigma^i P(0, 0) . \tag{7.9}$$

From Eq. 7.4 and Eq. 7.9, we have for the states (i, j) with $j = 1$:

$$P(i, 1) = \rho (P(i-1, 1) + \sigma^{i-1} P(0, 0)) . \tag{7.10}$$

Repeatedly substituting Eq. 7.10 into itself yields:

$$P(i, 1) = \rho^i P(0, 1) + \sum_{k=1}^{i} \rho^k \sigma^{i-k} P(0, 0) . \tag{7.11}$$

Using Eq. 7.5, this results, for all *i*, in:

$$P(i,1) = \left(\frac{\lambda}{r}\rho^i + \sum_{k=1}^{i} \rho^k \sigma^{i-k}\right) P(0,0) \quad . \tag{7.12}$$

With the help of the Cauchy product rule we now could obtain a closed-form expression for the steady state probability distribution of the Markov chain, by solving the normalization equation, Eq. 7.6, for $P(0,0)$. However, a more elegant solution is obtained when we realise that the long-term fraction of time a server is busy equals the utilization ρ, i.e., the following equality holds:

$$\sum_{i=1}^{\infty} P(i,1) = \rho \quad . \tag{7.13}$$

By substituting Eq. 7.13, Eq. 7.5, and Eq. 7.9 in the normalisation equation (Eq. 7.6), we obtain:

$$P(0,0)\left(\sum_{i=0}^{\infty} \sigma^i + \frac{\lambda}{r}\right) + \rho = 1 \quad . \tag{7.14}$$

Rewriting the geometric series yields:

$$P(0,0)\left(\frac{1}{1-\sigma} + \frac{\lambda}{r}\right) + \rho = 1 \quad , \tag{7.15}$$

which, after substituting Eq. 7.8, results in the following expression for $P(0,0)$:

$$P(0,0) = (1-\rho)\frac{1/\lambda}{1/\lambda + 1/r + 1/c} \quad . \tag{7.16}$$

Substitution of this expression in Eq. 7.9 and Eq. 7.12 leads to a closed-form solution for the stationary state probabilities.

Having deduced this closed-form solution we can derive expressions for a number of performance measures. For some of the measures, we make use of the Cauchy product rule ([30]), according to which:

$$\sum_{i=0}^{\infty}\sum_{k=0}^{i} \rho^k (i-k)\,\sigma^{i-k} = \left(\sum_{i=0}^{\infty} i\sigma^i\right)\sum_{i=0}^{\infty} \rho^k \qquad (\rho, \sigma < 1) \quad . \tag{7.17}$$

The following measures have been derived for the evaluation of the OCDR mechanism:

- Average number of packets in the system, $E[N]$:

$$E[N] = \sum_{i=0}^{\infty} i\,(P(i,0) + P(i,1)) = \frac{\rho}{1-\rho} + \left(\frac{\lambda + c}{c}\right)\frac{1/c}{1/\lambda + 1/r + 1/c}\ ; \qquad (7.18)$$

- Average delay, $E[T]$, using Little's law:

$$E[T] = \frac{1}{\lambda}E[N] = \frac{1}{\lambda}\left(\frac{\rho}{1-\rho} + \left(\frac{\lambda + c}{c}\right)\frac{1/c}{1/\lambda + 1/r + 1/c}\right)\ ; \qquad (7.19)$$

- The fraction of time a connection is available, $E[V]$, using Eq. 7.13:

$$E[V] = \sum_{i=0}^{\infty} P(i,1) = \rho + (1-\rho)\frac{1/r}{1/\lambda + 1/r + 1/c}\ ; \qquad (7.20)$$

- Average reserved bandwidth, $E[B]$, in bits/s:

$$E[B] = \mu l E[V] = \mu l\left(\rho + (1-\rho)\frac{1/r}{1/\lambda + 1/r + 1/c}\right)\ ; \qquad (7.21)$$

- Average number of connection setups per second, $E[S]$:

$$E[S] = c\sum_{i=1}^{\infty} P(i,0) = (1-\rho)\frac{1}{1/\lambda + 1/r + 1/c}\ . \qquad (7.22)$$

In Eq. 7.18 and Eq. 7.19, the first term equals the M/M/1 result for the average number of packets in the system and the average delay respectively. In Eq. 7.20, the first term (ρ) corresponds to the fraction of time that a connection is used for transmission. The second term expresses the time that an existing connection is idle.

7.2.3 OCDR under IPP Traffic

In order to model the behaviour of the OCDR mechanism with the IPP, described in Section 7.2.1, the CTMC of the previous section needs to be extended. The system is now modelled by the process $\{N(t), V(t), A(t)\}$, which is again a CTMC. Recall that $A(t)$ denotes the state of the arrival process at time t.

We are again interested in the steady-state behaviour of the CTMC. We define the steady state probabilities, $P(i,j,k)$, as follows:

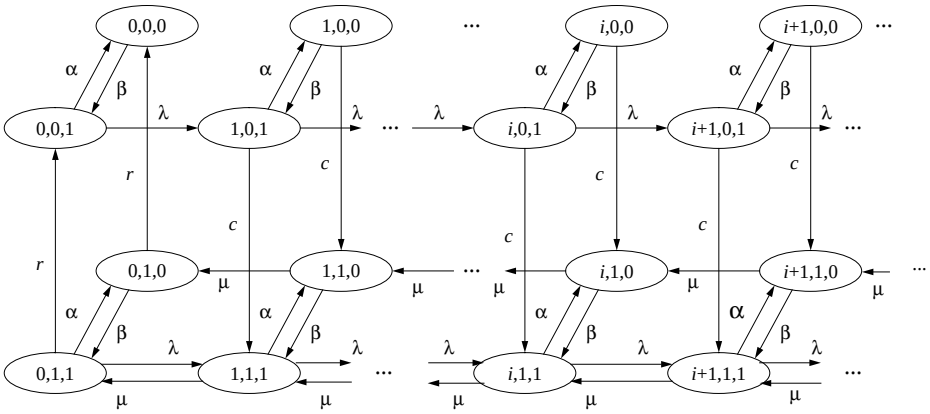


Figure 7-4: CTMC for OCDR under IPP Traffic

$$P(i,j,k) = \lim_{t \to \infty} P\{N(t) = i \wedge V(t) = j \wedge A(t) = k\} \quad . \tag{7.23}$$

The state transition diagram of this CTMC is depicted in Figure 7-4. It is similar to the one for Poisson arrivals (Figure 7-3). The state space is duplicated, to incorporate the state of the arrival process. The 'front plane' of states (the states labelled $(i, j, 1)$) represent the situation where the arrival process is in a burst. The transitions between the states are identical to those for Poisson arrivals. The 'back plane' of states (the states labelled $(i, j, 0)$) represent the situation where the arrival process is in an interburst period. It is identical to the 'front plane' except for the transitions with rate λ, which have been removed to represent the absence of arrivals. The system transits from the 'front plane' to the 'back plane' and back with rates α and β respectively.

For the analysis of this CTMC, we use the matrix geometric solution method developed by Neuts ([100]; see also [31] and [99]). Therefore, the repetitive structure of the CTMC is utilized. Let the states be ordered lexicographically ($(0, 0, 0)$, $(0, 0, 1)$, $(0, 1, 0)$, $(0, 1, 1)$, $(1, 0, 0)$, $(1, 0, 1)$, etc.). Then the generator matrix $\boldsymbol{Q}$, of the CTMC can be written as follows:

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{B}_{00} & \boldsymbol{B}_{01} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \cdots \\ \boldsymbol{B}_{10} & \boldsymbol{A}_1 & \boldsymbol{A}_0 & \boldsymbol{0} & \boldsymbol{0} & \cdots \\ \boldsymbol{0} & \boldsymbol{A}_2 & \boldsymbol{A}_1 & \boldsymbol{A}_0 & \boldsymbol{0} & \cdots \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{A}_2 & \boldsymbol{A}_1 & \boldsymbol{A}_0 & \cdots \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{A}_2 & \boldsymbol{A}_1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \tag{7.24}$$

where all constituting elements of $\boldsymbol{Q}$ are 4×4 matrices, and $\mathbf{0}$ is a matrix with all zeros. $\boldsymbol{Q}$ has a tridiagonal form, similar to the generator matrix of an M/M/1 queue. However, here the constituting elements are matrices themselves, unlike in the M/M/1 model, where the constituting elements are scalars. According to the matrix geometric solution technique, the stationary state probability distribution of the Markov process can be easily obtained after numerically solving a matrix quadratic equation with $\boldsymbol{A}_0$, $\boldsymbol{A}_1$, and $\boldsymbol{A}_2$ as coefficients. For this purpose, we have used the software tool Xmgm ([48]).

Let us now give the constituting matrices of the generator matrices. The set of states of the CTMC for which the number of customers in the system equals i is called a level i. Transitions between levels correspond to the arrival or departure of a packet. Transitions within a level correspond to a change in the state of the arrival process, or a change in the presence of the connection. The matrix $\boldsymbol{A}_0$ gives the transition rates between states, given that the system goes from level i to level $i+1$. It is defined as follows:

$$\boldsymbol{A}_0 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \lambda & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda \end{bmatrix}. \tag{7.25}$$

$\boldsymbol{A}_2$ describes the transitions between states, while the system transits from level i to level $i-1$:

$$\boldsymbol{A}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & \mu \end{bmatrix}. \tag{7.26}$$

Finally, $\boldsymbol{A}_1$ describes the transitions within a level, and on the diagonal, the zero-row compensation of the generator matrix:

$$\boldsymbol{A}_1 = \begin{bmatrix} -(\beta + c) & \beta & c & 0 \\ \alpha & -(\alpha + c + \lambda) & 0 & c \\ 0 & 0 & -(\beta + \mu) & \beta \\ 0 & 0 & \alpha & -(\alpha + \lambda + \mu) \end{bmatrix}. \tag{7.27}$$

The $\boldsymbol{B}$-matrices define the boundary transitions, i.e., the transitions to and/or from level 0. It turns out that for the CTMC described in this subsection, the transitions to and from level 0 are the same as for other levels, i.e.,

$$\boldsymbol{B}_{01} = \boldsymbol{A}_0 \text{ , and} \tag{7.28}$$

$$\boldsymbol{B}_{10} = \boldsymbol{A}_2 \text{ .} \tag{7.29}$$

The transition within level 0 are defined as follows:

$$\boldsymbol{B}_{00} = \begin{bmatrix} -\beta & \beta & 0 & 0 \\ \alpha & -(\alpha+\lambda) & 0 & 0 \\ r & 0 & -(\beta+r) & \beta \\ 0 & r & \alpha & -(\alpha+\lambda+r) \end{bmatrix} . \tag{7.30}$$

7.2.4 OCDR with Erlang Holding and Connection Setup Times

Up to now, we have modelled the holding time of the OCDR mechanism as an exponentially distributed time. In the real system, this time will probably be determined by a fixed timer value, and hence be a deterministic one. In order to model this holding time more accurately, and to check our previous model, we will now model it as an Erlang n distribution, i.e., a distribution with n identical exponentially distributed stages. It is known that the squared coefficient of variation of such a distribution converges to 0 if $n \to \infty$, i.e., the Erlang distribution approaches a deterministic distribution ([84]). Similarly, we model the connection setup time with an Erlang m distribution, since this time can also be assumed to have a lower variance than an exponential distribution.

Let us now enhance the model that we have developed for OCDR under IPP arrivals, in order to cope with the Erlang distributions for the holding time and the connection setup time. The system is again modelled with the CTMC $(N(t), V(t), A(t))$, but now the stochastic process $V(t)$ is defined to represent the stage of the Erlang distribution for the holding time or connection setup time. For an empty system, i.e., $N(t) = 0$, $\{V(t), t \geq 0, V(t) \in \{0, 1, \ldots, n\}\}$ defines the stage of the holding time, which is initially n. Assuming the system stays empty, $V(t)$ decreases with a rate nr. If $V(t) = 0$ the connection has been released, otherwise it is still available. If the system becomes non-empty, i.e., $N(t) > 0$, before the connection has been released, the timer is reset, i.e., $V(t)$ starts at n again the next time the system becomes empty. For a non-empty system, i.e., $N(t) > 0$, $\{V(t), t \geq 0, V(t) \in \{0, 1, \ldots, m\}\}$ defines the stage of the

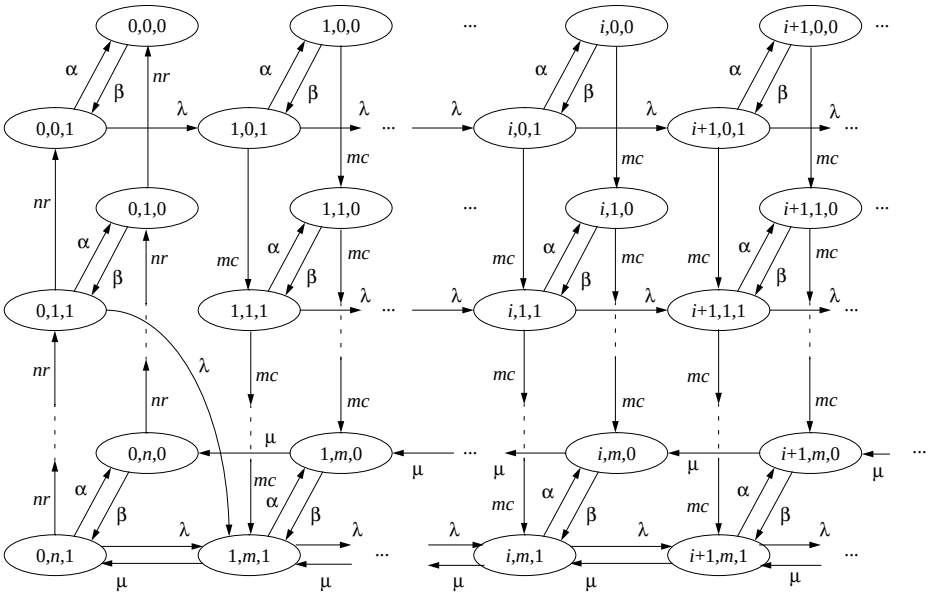


Figure 7-5: CTMC of OCDR with Erlang Holding and Connection Setup Times

connection setup time, which is initially 0. Now, $V(t)$ increases with a rate mc. If $V(t) = m$, the connection has been set up, otherwise it is not yet available. The stationary state probability distribution of the CTMC is still defined as in Eq. 7.23. However, note that the domain of $V(t)$ has changed.

The state transition diagram of the CTMC is shown in Figure 7-5. Its generator matrix $\boldsymbol{Q}$ has the same shape as the one in Eq. 7.24. However, the constituting matrices are defined differently. $\boldsymbol{A}_0$, $\boldsymbol{A}_1$, and $\boldsymbol{A}_2$ are matrices of size $2(m+1) \times 2(m+1)$, which are defined as follows:

$$\boldsymbol{A}_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & \lambda & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \lambda & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & \lambda \end{bmatrix}, \tag{7.31}$$

$$\boldsymbol{A}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & . & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & \mu & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & \mu \end{bmatrix}, \text{and} \tag{7.32}$$

$$\boldsymbol{A}_1 = \begin{bmatrix} -(\beta + mc) & \beta & mc & 0 & \dots & 0 & 0 \\ \alpha & -(\alpha + mc + \lambda) & 0 & mc & \dots & 0 & 0 \\ 0 & 0 & -(\beta + mc) & \beta & \dots & 0 & 0 \\ 0 & 0 & \alpha & -(\alpha + mc + \lambda) & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & . & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & -(\beta + \mu) & \beta \\ 0 & 0 & 0 & 0 & \dots & \alpha & -(\alpha + \lambda + \mu) \end{bmatrix}. \tag{7.33}$$

$\boldsymbol{B}_{01}$ is a matrix of size $(n+1) \times (m+1)$:

$$\boldsymbol{B}_{01} = \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & \lambda & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & \lambda \\ \vdots & \vdots & \vdots & \vdots & . & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & \lambda \end{bmatrix}. \tag{7.34}$$

$\boldsymbol{B}_{10}$ is a matrix of size $(m+1) \times (n+1)$, which is defined similar to $\boldsymbol{A}_2$:

$$\boldsymbol{B}_{10} = \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & . & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & \mu & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & \mu \end{bmatrix}, \tag{7.35}$$

Finally, $\boldsymbol{B}_{00}$ is a matrix of size $(m+1) \times (m+1)$, which is defined as

$$\boldsymbol{B}_{00} = \begin{bmatrix} -\beta & \beta & 0 & 0 & \dots & 0 & 0 \\ \alpha & -(\alpha+\lambda) & 0 & 0 & \dots & 0 & 0 \\ nr & 0 & -(\beta+nr) & \beta & \dots & 0 & 0 \\ 0 & nr & \alpha & -(\alpha+\lambda+nr) & \dots & 0 & 0 \\ : & : & : & : & . & : & : \\ 0 & 0 & 0 & 0 & \dots & -(\beta+nr) & \beta \\ 0 & 0 & 0 & 0 & \dots & \alpha & -(\alpha+\lambda+nr) \end{bmatrix} . \quad (7.36)$$

Now that we have defined the constituting matrices of the generator matrix, we are able to solve for the stationary state probabilities of the CTMC, again using Xmgm. From the obtained probabilities we can derive a number of performance measures:

- Average number of packets in the system, $E[N]$:

$$E[N] = \sum_{i=1}^{\infty} i \sum_{j=0}^{m} (P(i,j,0) + P(i,j,1)) \; ; \quad (7.37)$$

- Average delay, $E[T]$, using Little's law:

$$E[T] = \frac{1}{E[A]\lambda} E[N] \; ; \quad (7.38)$$

- The fraction of time a connection is available, $E[V]$:

$$E[V] = \sum_{i=1}^{\infty} (P(i,m,0) + P(i,m,1)) + \sum_{j=1}^{n} (P(0,j,0) + P(0,j,1)) \; ; \quad (7.39)$$

- Average reserved bandwidth, $E[B]$:

$$E[B] = \mu l E[V] \; ; \quad (7.40)$$

- Average number of connection setups per second, $E[S]$, by recognizing that the establishment of a connection needs to be performed each time a packet arrives while the system is empty and no connection is available:

$$E[S] = \lambda P(0,0,1) \; . \quad (7.41)$$

Note that this model is equivalent to the CTMC of the previous section for $n = 1$ and $m = 1$. The expressions for the performance measures (Eq. 7.37 to Eq. 7.41) apply also to that model if n and m are substituted by 1.

7.3 Evaluation

Using the models described in Section 7.2.2, Section 7.2.3, and Section 7.2.4, we will evaluate the performance of the OCDR mechanism. This will be done by comparing it to the CpP and PC mechanisms, with respect to the performance measures identified in Section 7.1.3.

In the following, we first give the values for the model parameters that have been used, in Section 7.3.1. Then, in Section 7.3.2, we evaluate the performance of OCDR with a workload of Poisson traffic. In Section 7.3.3, we evaluate OCDR with a workload of bursty (IPP) traffic. In Section 7.3.4, we do the same, now assuming that the holding time and connection setup time have an Erlang distribution. In the last three subsections, we continue to assume bursty traffic and Erlang holding time and connection setup time. In Section 7.3.5, we analyse the behaviour of OCDR under varying holding time, in order to determine the optimal value for this control parameter. Finally, in Section 7.3.6 and Section 7.3.7, we evaluate the performance of OCDR under varying load and burst length respectively.

7.3.1 Parameter Values

The values for the model parameters, given in this subsection, are the default parameters used in the experiments. Unless stated differently, these values are used. They are summarized in Table 7-2. Values may be different for different models. Furthermore, sometimes no default value is assumed for a parameter (-), or a parameter is not applicable (n.a.) to a model because it is not defined for that model.

The workload parameters for the IPP traffic are based on measurements in [83], taking into account the high-speed character of future applications. The average interburst time has been taken to be 25 seconds, i.e., $\beta = 0.04$. The average burst time is 1 second, i.e., $\alpha = 1$. The average number of arrivals per burst is 100, i.e., $\lambda = 100$. This is an order of magnitude higher than the value measured in [83], to reflect the expected increase in traffic intensity in the future.

For Poisson traffic, we only have to define the parameter of the exponential distribution of the interarrival time, λ. In order to have an average number of arrivals per second equivalent for IPP traffic and Poisson traffic, we have adopted $\lambda = 100E[A] = 100\beta/(\alpha+\beta) = 100/26$ for Poisson traffic.

	Poisson Traffic	IPP Traffic (Exponential Holding and Connection Setup Times	IPP Traffic (Erlang Holding and Connection Setup Times
λ (s^{-1})	100/26	100	100
μ (s^{-1})	-	-	125
r (s^{-1})	-	-	10
c (s^{-1})	10	10	10
l (bits)	10 000	10 000	10 000
α (s^{-1})	n.a.	1	1
β (s^{-1})	n.a.	0.04	0.04
n ()	n.a.	n.a.	30
m ()	n.a.	n.a.	5

Table 7-2: Default Values for Model Parameters

The average packet length, l, is assumed to be 10 kbits for both Poisson and bursty traffic.

The average of the connection setup time, $1/c$, is assumed to be 100 ms, i.e., $c = 10$. The default value for the rate at which packets are served if a connection is available has been chosen such that the utilization during a burst is 0.8, i.e., $\mu = 125$. Furthermore, experiments (see e.g., Figure 7-9) have suggested a holding time of 0.1 second ($r = 10$) as an optimal value.

In the model with Erlang holding and connection setup times, the number of stages for the holding time has been taken to be 30. The connection setup time has an Erlang 5 distribution.

7.3.2 Poisson Traffic

The expected gain of the OCDR mechanism is that it claims less resources from the ATM network than a PC mechanism, because an outgoing connection is released during periods in which no traffic arrives. The average reserved bandwidth ($E[B]$) is a good measure to express this claim, i.e., it is an indication of the costs of the use of the ATM network. A disadvantage of the use of OCDR, compared to PC, is that some packets will experience a higher delay, because a connection must be established before they can be transferred. However, by serving packets at a high rate (μ), if the connection is available, i.e., by requesting

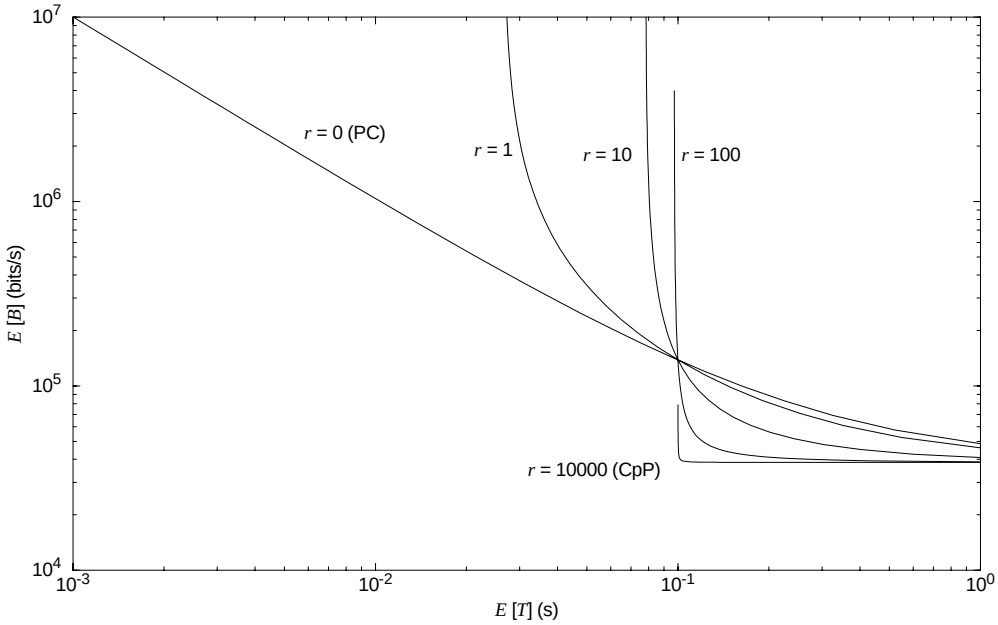


Figure 7-6: Average Reserved Bandwidth vs. Average Delay (Poisson Traffic)

a high bandwidth for the connection, the average delay ($E[T]$) can be kept acceptable.

The problem we are interested in, is the following. Given a certain load (λ), and a certain required average delay ($E[T]$), what holding time ($1/r$) and what connection bandwidth (μ) should be chosen to achieve a minimal average reserved bandwidth ($E[B]$), i.e., minimal costs?

In order to deal with this problem, we investigate how the obtained average reserved bandwidth relates to the obtained average delay, for different values of the connection bandwidth (μ). In Figure 7-6, we display both measures for varying μ, i.e., the curves that are drawn in the figure are parametric curves. μ is increasing from right to left, i.e., the average delay decreases, and the average reserved bandwidth increases with increasing μ. Curves are depicted for different holding times $1/r$. Note that an infinite holding time ($r = 0$) corresponds to the PC mechanism, and that a zero holding time (approached by $r = 10000$) corresponds to the CpP mechanism.

Let us discuss the characteristics of the graph in more detail. All the curves with $r > 0$ converge to some vertical asymptote, i.e., they show an asymptotic behaviour to a limiting value of $E[T]$. This is the limiting behaviour for $\mu \to \infty$, for which Eq. 7.19 reduces to

$$\lim_{\mu \to \infty} E[T] = (1/c) \frac{(1/\lambda + 1/c)}{(1/\lambda + 1/r + 1/c)} . \quad (7.42)$$

In this expression, $1/c$ is the expected time a customer has to wait if no connection is available upon arrival. The remaining factor can be considered as the probability that no connection is available for an arriving customer. We directly see that this implies that for the CpP mechanism ($r \to \infty$) the limiting value of the average delay equals $1/c$, the connection setup time, since all packets will find no connection available. For $r = 0$, the limit of $E[T]$ is 0 for $\mu \to \infty$, which implies that for the PC mechanism every delay demand can be guaranteed. However, due to the fact that μ is finite because of the finite capacity of an ATM link, this limit of 0 is not reached.

From Eq. 7.19 and Eq. 7.21 can be derived that the curves cross at $E[T] = 1/c$. Note that in this point the values for μ differ for the various curves. Only the average reserved bandwidth, which is the product of μ, the average packet length (l), and the fraction of time a connection is available (Eq. 7.20) is constant.

We see from the curves that the optimal value for r for some average delay is either $r = 0$ when the required average delay is smaller than 0.1 s, or $r \to \infty$ when the required average delay is larger than 0.1 s. Since the required average delay can be expected to be in the order of 0.01 s, the PC mechanism ($r = 0$) is the only suitable mechanism, if traffic arrives according to a Poisson process.

Concluding we can state that the OCDR mechanism is not advantageous if packets arrive to a CLS or end-system according to a Poisson arrival process. Either the CpP or PC mechanism needs the least bandwidth to fulfil some delay demand. This is due to the fact that the Poisson process does not exhibit any correlation between subsequent arrivals. In the following subsections we will evaluate the OCDR mechanism for an IPP in which this correlation does exist.

7.3.3 IPP Traffic with Exponential Holding and Connection Setup Times

Results for the OCDR model assuming arrivals according to an IPP and exponential holding and connection setup times are obtained using Xmgm with Eq. 7.25 through Eq. 7.30 as input. Analogously to the previous subsection, we plot the average reserved bandwidth versus the average delay (see Figure 7-7). Curves have been drawn for the same values of r. Again, the varying parameter of the parametric curves is μ.

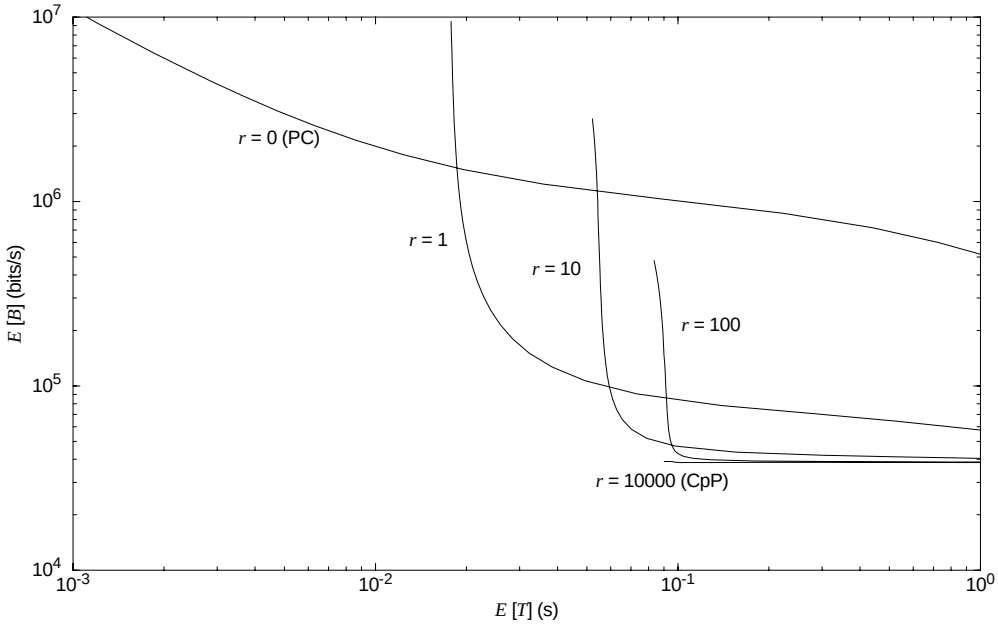


Figure 7-7: Average Reserved Bandwidth vs. Average Delay (IPP, n=1, m=1)

Contrary to the OCDR mechanism under Poisson traffic, the mechanism is advantageous if packets arrive to the CLS or end-system according to an IPP with the given parameters. Depending on the required average node delay, one of the values for r yields the lowest average reserved bandwidth. It can roughly be said that a PC mechanism ($r = 0$) is the optimal solution for a required average delay of less than 0.01, and a CpP mechanism ($r \to \infty$) for a required average delay of more than 0.1. For intermediate values, other values for r are optimal.

7.3.4 IPP Traffic with Erlang Holding and Connection Setup Times

In the graph of Figure 7-7, we have still assumed that the holding time and the connection setup time are exponentially distributed. In a real system this will definitely not be true. The holding time will be a deterministic one, and the connection setup time will also have a lower variance than an exponential distribution.

In order to model the real system more accurately, we assume that both the holding time and the connection setup time are distributed according to an Erlang distribution. To reflect the fact that the holding time will be deterministic. we model it with an Erlang 30 distribution, i.e., $n = 30$ in Figure 7-5. For the connection setup time, we assume an Erlang 5 distribution, i.e., $m = 5$. In [31], it is shown that the results from the model are not very sensitive to the exact value

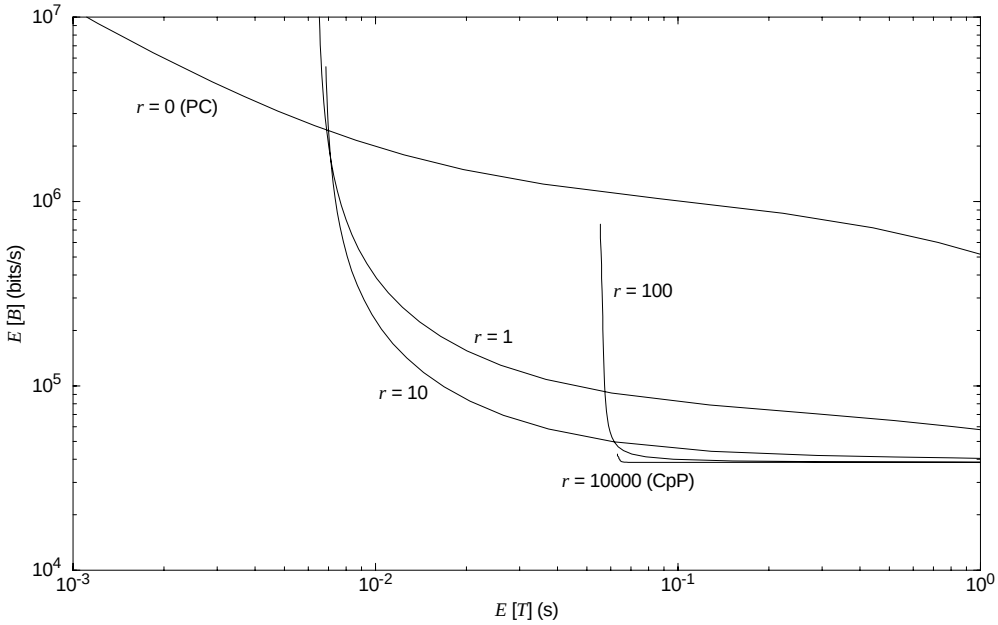


Figure 7-8: Average Reserved Bandwidth vs. Average Delay (IPP, n=30, m=5)

of *m* and *n*, in the range of the system parameters that have been used. The mentioned distributions are used throughout the rest of the section.

Let us again give the same type of graph as in the previous two subsections. Figure 7-8 gives curves of the average reserved bandwidth versus the average delay. The most conspicuous difference between this graph and the previous one appears in the curve for $r = 10$. This curve has shifted to the left, i.e., a lower average delay can be provided with the same average reserved bandwidth. As a result, OCDR, with a holding time of 0.1 s, is the most optimal mechanism for a wide range of delay requirements.

Why is a holding time of 0.1 s the most optimal one? If the holding time is (an order of magnitude) lower (e.g., $r = 100$), it is frequently shorter than the packet interarrival time during a burst. Consequently, the connection is released for a very short time, because the next packet of the burst will arrive shortly, and hence, only little bandwidth is saved at the cost of an increased average delay. If the holding time is (an order of magnitude) higher (e.g., $r = 1$), it is of the same order as the length of the entire burst, and bandwidth is wasted because the connection is not released fast enough. Now, the difference between the results for exponential and Erlang holding time become also clear. If the holding time would be taken from an exponential distribution instead of an Erlang (or deterministic) one, its actual value would often be far beside the mean of the distribu-

tion. As a result, the OCDR mechanism would perform worse, despite the optimal mean holding time.

For a required average delay in the range between 0.007 s and 0.06 s, the OCDR mechanism with a holding time of 0.1 s can fulfil the delay requirements at the lowest cost, i.e., with the lowest average reserved bandwidth. This range can expected to be the operational region of the mechanism. Compared to the PC mechanism a significant reduction of the average reserved bandwidth can be obtained, i.e., up to 95% (for $E[T] = 0.06$).

7.3.5 Optimal Holding Time

From now on, we assume IPP traffic and Erlang holding and connection setup times. In order to obtain more insight in the optimal value for the connection holding time, we do an experiment where we vary this parameter, keeping the other parameters constant (see Table 7-2 for the default values). We perform the experiment for different values of μ, so that the utilization during a burst is between 0.5 ($\mu = 200$) and 1.0 ($\mu = 100$). Here we only show a graph for the average number of connection setups per second ($E[S]$) as a function of the holding time ($1/r$), in Figure 7-9.

The influence of the holding time on the behaviour of the OCDR mechanism becomes very clear from the graph. If the holding time is larger than 0.1 second, the rate at which new connections are established is very close to the rate at which new bursts start, i.e., once per 26 seconds. This indicates the desired behaviour of OCDR. The connection is released between bursts, and held during bursts. If the holding time becomes much larger, the average number of connection setups per second decreases, because the connection will no longer be released between bursts. The mechanism behaves like a PC mechanism. For small holding times (smaller than 0.01 second), the rate at which connections are set up depends on the utilization during a burst. In general, this rate is high, because the connection is often released during a burst. If the utilization is high ($\mu = 100$), this is not the case because the system will only rarely become empty during a burst. In the range of $1/r$ between 0.01 s and 0.1 s, the average number of connection setups per second is sharply decreasing with increasing holding time. Concluding, we can state that for the given parameters, a holding time of 0.1 second is optimal for the proper operation of the OCDR mechanism.

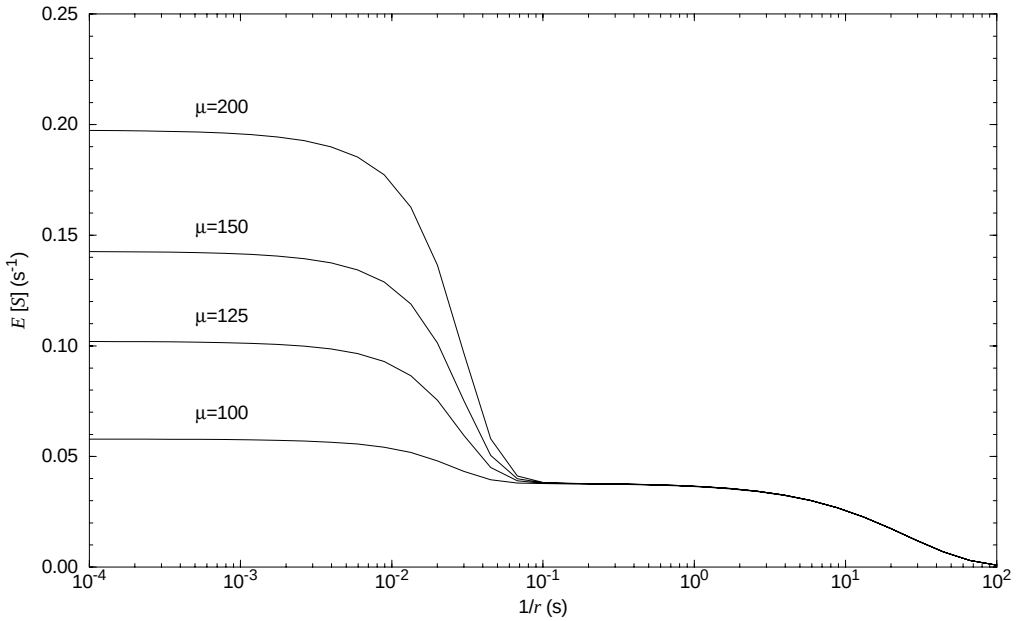


Figure 7-9: Average Number of Connection Setups per Second for Varying Holding Time

7.3.6 Behaviour under Varying Load

In Section 7.3.4, we have compared the OCDR mechanism to the PC and CpP mechanisms. It turned out that the OCDR mechanism can reduce the average reserved bandwidth up to 95%, depending on the required average delay. Of course, the obtained results depend on the parameters of the arrival process. In order to obtain insight in this dependency, the effect of the load and the burst length on the results will be investigated in the following subsections.

In order to vary the load on the connectionless protocol, λ is varied. Increasing λ means that the interarrival time in a burst decreases, and the number of arrivals per burst increases. μ is kept constant at 125, which implies that the bandwidth reserved for a connection is 1.25 Mbits/s (μl), if the connection is established. We give curves for three values of r. A holding time of 0.1 seconds ($r = 10$) turned out to be most optimal for OCDR in the previous subsection. For comparison, we also display curves for $r = 0$ (PC mechanism) and $r = 10000$ (approximating the CpP mechanism). Graphs are given for the average delay, $E[T]$ (Figure 7-10), the average reserved bandwidth, $E[B]$ (Figure 7-11), and the average number of connection setups per second, $E[S]$ (Figure 7-12).

From Figure 7-10, it can be observed that for the same μ, the average delay for OCDR is about twice the delay for PC if λ is between 50 and 125. This range corresponds to a utilization of the connection during a burst between 0.4 and 1.0.

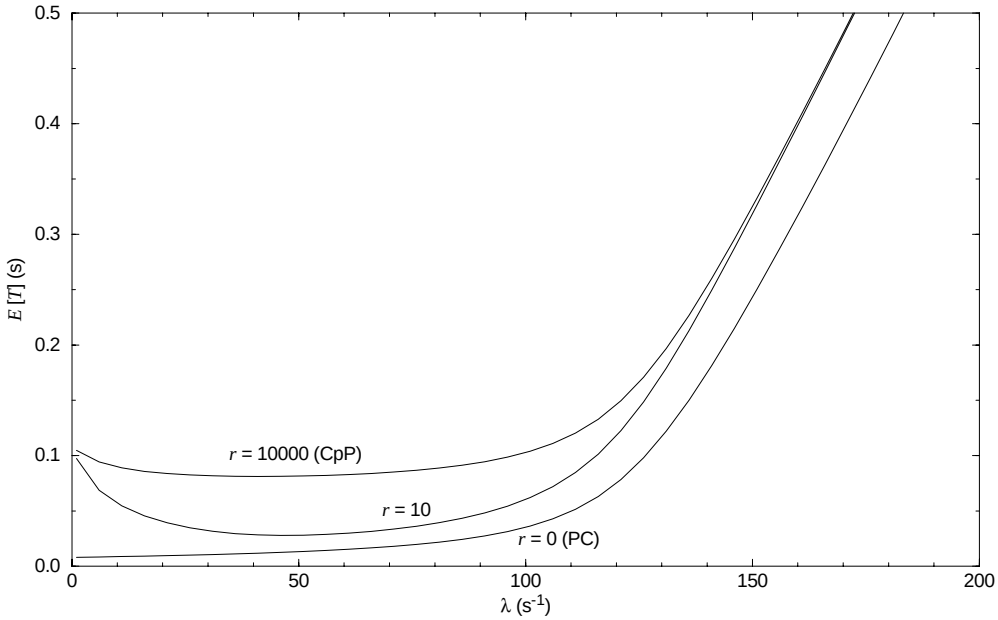


Figure 7-10: Average Delay under Varying Load

Recall that λ is the arrival rate during a burst. The CpP mechanism has a much higher delay here, because the connection is often released during a burst. The same can be observed for the OCDR mechanism if λ decreases below 50. If the load approaches zero, the mean delay for both the CpP and the OCDR mechanism approaches the sum of the packet transmission time and the connection setup time ($1/\mu + 1/c = 0.108\,\text{s}$), because a new connection needs to be setup for every packet. Here, the mean delay of the PC mechanism becomes the packet transmission time ($1/\mu = 0.008\,\text{s}$), because a packet will not experience waiting time any more.

For $\lambda > \mu$ the average delays for OCDR and CpP converge, because the system will no longer become empty during a burst, due to temporary overload. Consequently, the CpP mechanism will not release the connection during the burst. From $\lambda = 150$, the difference in delay between the mechanisms will be constant with increasing load. The delay of the mechanisms will increase almost linearly. If the load is so high that the system cannot transmit the excess traffic of a burst in an interburst period any more, the delay will increase more rapidly with increasing load.

In Figure 7-11, it can be seen that the average reserved bandwidth of the OCDR mechanism is almost constant for $\lambda > 50$. This indicates that the OCDR mechanism is stable with these parameters: the connection is available during bursts,

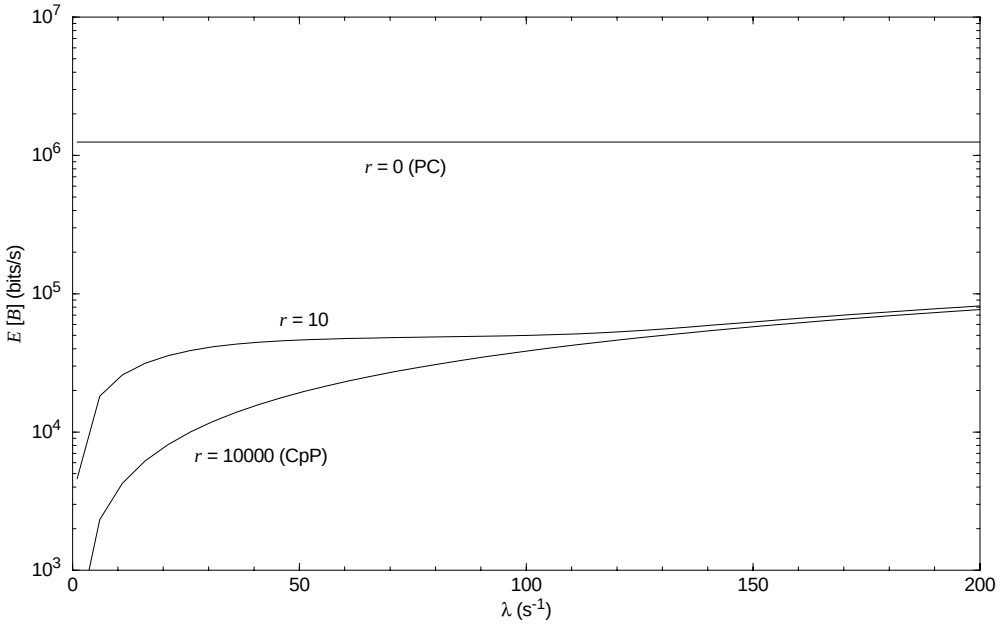


Figure 7-11: Average Reserved Bandwidth under Varying Load

and released between bursts, regardless of the traffic intensity during the burst. Of course, the average reserved bandwidth in case of a permanent connection (PC) is constant at 1.25 Mbits/s. For the CpP mechanism, only the bandwidth that is really used is reserved, i.e., $E[B] = E[A]\lambda l = \lambda/2600$ Mbits/s (see Eq. 7.1). The mechanism does not waste any bandwidth. As a result, the difference between the curve for CpP and the curves for the other mechanisms can be interpreted as the bandwidth that is wasted by the concerned mechanisms. If λ is larger than μ, i.e., $\lambda > 125$, OCDR wastes only very little bandwidth. The PC mechanism on the other hand uses about 20 times as much.

For high loads, a small difference between the average reserved bandwidth of the CpP and OCDR mechanisms remains, because of the bandwidth reserved during the holding times, after each burst. When the system becomes overloaded, i.e., $\lambda E[A] \rightarrow \mu$, the average reserved bandwidth for both mechanisms converges to the one for the PC mechanism, because the connection will not be released any more. If the load approaches zero, the reserved bandwidth for the CpP mechanism goes to zero. The average reserved bandwidth for the OCDR mechanism goes to zero as well, but much slower, because a connection, established for the transfer of a single packet, will only be released after the holding time expires.

Figure 7-12 shows the average number of connection setups per second ($E[S]$) as a function of the load (λ). Clearly, $E[S] = 0$ for the PC mechanism. The shape of

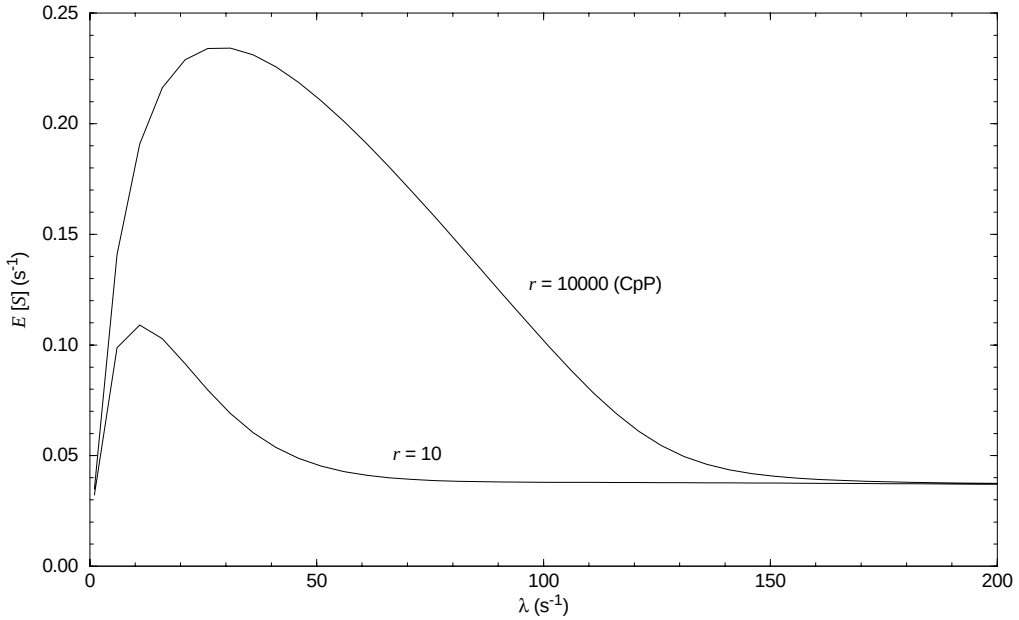


Figure 7-12: Average Number of Connection Setups per Second under Varying Load

the curves for OCDR and CpP is explained as follows. If the load is close to zero, $E[S]$ equals $\lambda/26$, since a new connection is established for each packet. If the load increases, more packets can be served by the same connection, because they arrive before the connection is released. Consequently, the average number of connection setups per second increases less than proportional with the load. For certain λ, the increase of the number of arriving packets is cancelled by the increase of the number of packet served per connection, so that the curve has its peak. From this point onwards, $E[S]$ decreases for increasing λ. If the load is such that all packets of a burst are served by a single connection, $E[S]$ will not decrease any more. It will be close to the rate at which new bursts are started ($1/(1/\alpha + 1/\beta) = 1/26$). When the system becomes overloaded, i.e., $\lambda E[A] \rightarrow \mu$, the average number of connection setups per second approaches zero, because the connection will not be released any more.

The load from which $E[S]$ remains constant differs for the CpP and OCDR mechanisms. For CpP, it remains constant if λ increases beyond 150, i.e., if the load during the burst exceeds the capacity. For OCDR, $E[S]$ remains constant if λ increases beyond 50, i.e., if the interarrival time during a burst exceeds the holding time most of the time. This confirms the indication that OCDR (with $r = 10$) operates properly if $\lambda > 50$, since in this range, a single connection is used to transmit an entire burst.

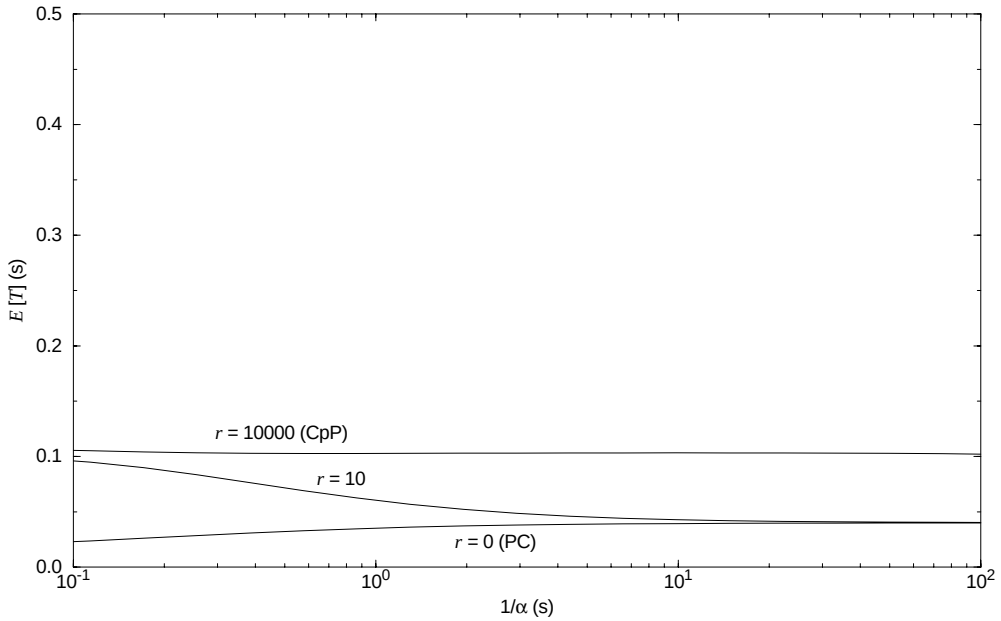


Figure 7-13: Average Delay under Varying Burst Length

Concluding, we can state that the OCDR mechanism performs well with the given parameters if the utilization of a connection during a burst is between 0.4 and 1.0. For higher utilizations, the average delay becomes too high for all mechanisms. For lower utilizations, OCDR does not perform well, resulting in a relatively high average delay, and a high load on the signalling system.

7.3.7 Behaviour under Varying Burst Length

Next, we want to examine the behaviour of the OCDR mechanism under varying burst length ($1/\alpha$). Note that the burst length is expressed in time, not in the number of packets. Since the arrival rate during a burst (λ) is kept constant at 100, the average number of packet per burst equals $100/\alpha$. In [83], burst lengths of tens of packets have been measured, but the increasing volumes of data that need to be transported for applications may lead to larger bursts. A burst length of $1/\alpha = 1$ seems to be a good indication of a realistic value. The interburst time ($1/\beta$) is varied proportional to the burst length, so that the fraction of time the arrival process is in the burst state ($E[A]$), and hence the mean arrival rate is constant. Again, graphs for the average node delay $E[T]$ (Figure 7-13), the average reserved bandwidth $E[B]$ (Figure 7-14), and the average number of connection setups per second $E[S]$ (Figure 7-15) are given.

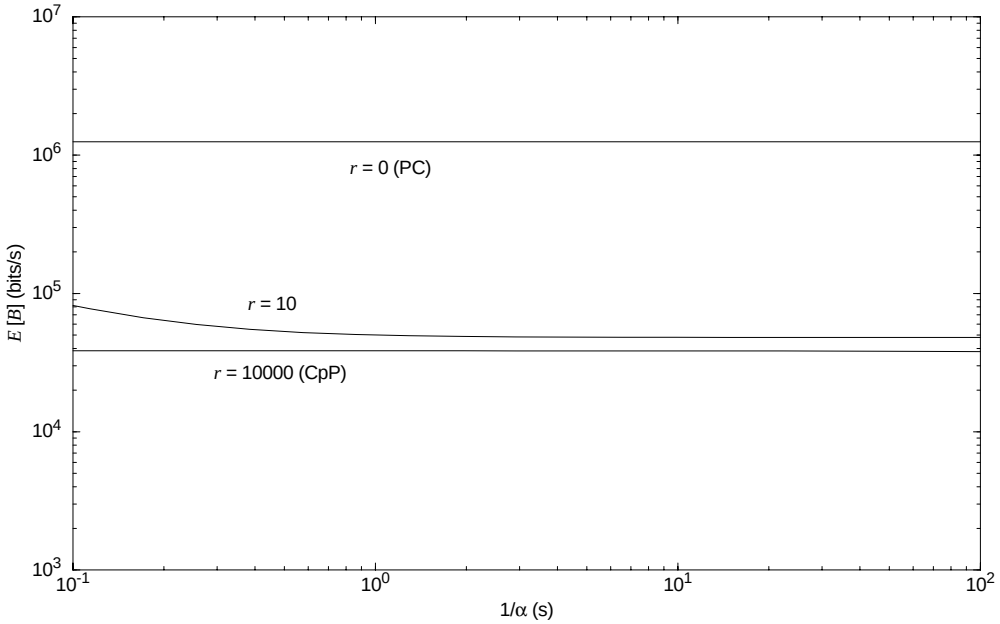


Figure 7-14: Average Reserved Bandwidth under Varying Burst Length

From Figure 7-13, it can be observed that the average delay of the CpP mechanism is almost constant under varying burst length. It is constantly close to the sum of the average setup time and the average transmission time ($1/c + 1/\mu = 0.108$ s). The average delay of the PC mechanism increases only slightly with the burst length. For very short burst lengths (out of the range of the graph), the average delay will approach the average transmission time ($1/\mu$). For increasing burst lengths, the average delay will convert to the average delay with a regular Poisson arrival process ($(1/\mu) / (1 - \lambda/\mu) = 0.04$), since the interburst periods will not contribute to the measure any more.

The average delay of the OCDR mechanism converges to the average delay of the PC mechanism if the burst length increases. The effect of the extra delay for establishing a connection decreases, because this is needed for a decreasing fraction of the customers. For decreasing burst lengths, the average delay of OCDR approaches the delay of the CpP mechanism since the correlation between the subsequent interarrival times disappears.

The average reserved bandwidth for OCDR is not very dependent on the burst length. In Figure 7-14, it can be seen that it only increases slightly if the burst length decreases. The increase is caused by the fact that the connection is established and released more often. Before the connection is released, it has not been used for the holding time, i.e., bandwidth is wasted. Again, the reserved band-

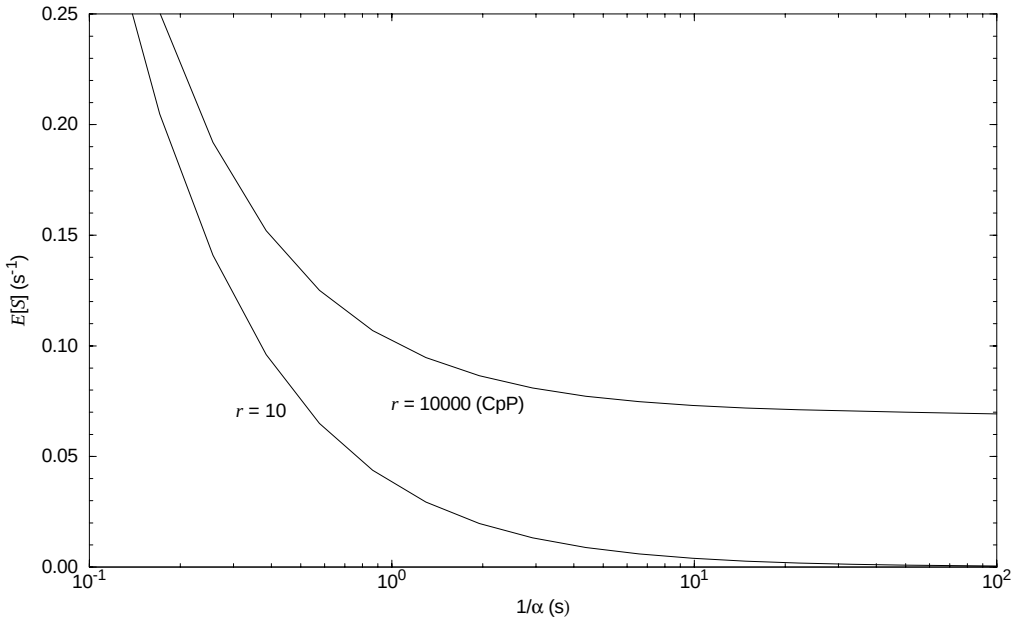


Figure 7-15: Average Number of Connection Setups per Second under Varying Burst Length

width for the PC mechanism is fixed at 1.25 Mbits/s. The average reserved bandwidth for the CpP mechanism equals $E[A]\lambda l = 1/26$ Mbits/s, i.e., the number of bits per second that are offered to the system.

Figure 7-15 indicates that the average number of connection setups per second is very close to the average number of bursts that start per second ($\alpha/26$). It is slightly higher, because the connection is sometimes accidentally released during a burst. For the CpP mechanism, $E[S]$ is much higher, since the connection is often released during a burst. Both curves converge if the burst length decreases, because having a connection per burst is the same as having a connection per packet if the burst consists of only a single packet. Clearly, the average connection setup rate equals zero for the PC mechanism.

From the graphs that we have considered, we can conclude that the average delay and the average reserved bandwidth of the CpP and PC mechanisms is hardly sensitive to the burst length. For each measure, the OCDR yields a value between those for the other two mechanisms.

7.4 Summary and Concluding Remarks

In this section, the connection management function has been defined as the function in a CLS or end-system that interacts with the signalling system to ensure that AAL connections for the transfer of packets are available when needed. A number of mechanisms have been identified that can be used to implement this function. The ‘On-demand Connection with Delayed Release’ mechanism is one that maintains a connection only during periods in which packets arrive regularly. If no packets have arrived after the last transmission for a period of time, called ‘holding time’, the connection is released.

In order to evaluate the performance of the OCDR mechanism, a number of models have been constructed and analysed. The mechanism has been compared with a ‘Permanent Connection’ and a ‘Connection per Packet’ mechanism. From the evaluation of a model that assumed a Poisson arrival process, some characteristic behaviour of the mechanism has been identified. However, this model did not reveal any advantages for the OCDR mechanism, because the underlying assumption of correlation in the arrival stream was not captured in the arrival process. Therefore, a second model was used that does capture this correlation. It assumed an IPP as the arrival process. The evaluation of this model showed that OCDR can support the same average delay with significantly less reserved bandwidth than the PC mechanism.

In order to model the deterministic holding time more accurately, a third model has been evaluated, assuming an Erlang-30 distribution for the holding time. In this model, it is also assumed that the connection setup delay is distributed according to an Erlang-5 distribution. The evaluation of this model reveals the same bandwidth reductions as the previous model. However, here the most optimal value for the holding time can be identified more clearly.

Compared to the PC mechanism, the average delay increases only slightly for a given bandwidth of the outgoing connection, if the connection is released after each burst. However, the reduction of the average reserved bandwidth is significant (about 95% for the used parameters). This behaviour can be obtained at the cost of extra signalling traffic, at the beginning and end of every burst. The evaluation shows that the advantage is most pronounced if the burst size is large. Furthermore, it is shown that the proper operation of the mechanism depends on the arrival rate during a burst. For the given parameters, the OCDR mechanism operates well if the packet arrival rate in a burst is between 40 and 100% of the packet transmission rate. For lower loads, the connection is released too often,

also within bursts, and hence the average number of connection setups per second and the average delay increase. For higher loads, none of the mechanisms operates well, because the system becomes overloaded during a burst, and the average delay becomes too high.

The major conclusion of the evaluation is that the OCDR mechanism is advantageous if the offered traffic has a bursty nature. The mechanism only works if interburst periods are long enough. This is probably not the case on the connections between CLSs, since traffic from a lot of applications will have to be sent over the same connection. As a result, there will probably not be long interburst periods, where no packets have to be transmitted. The OCDR mechanism is most suitable for the indirect method of operation, i.e., for connections between end-systems, and for the connections between end-systems and CLSs. In these cases, only packets from or to a single or a few application processes have to be supported by a single connection, so that the burst-silence cycle is most pronounced.

Look where all this talking got us, baby.

Live - Throwing Copper

Chapter 8
Conclusions

In this dissertation, the design of a system for broadband connectionless communications using the Asynchronous Transfer Mode has been addressed. The objective was to provide insight in some important aspects of the design, and to analyse design alternatives with respect to effectiveness, availability, scalability, and in particular, with respect to performance.

This last chapter reviews the major achievements of this dissertation, and presents directions for further work. For a more detailed discussion of the results of the individual chapters, we refer to their concluding sections.

In Section 8.1, we discuss the main conclusions and results. In Section 8.2, we indicate areas for further research.

8.1 Main Conclusions and Results

A number of applications that have to be supported by the B-ISDN have a connectionless nature. Furthermore, nowadays LANs, which ask for interconnection, use a connectionless protocol at the network layer (IP). Therefore, there is a clear demand for the provision of a connectionless service by the B-ISDN. Because ATM, the underlying switching and multiplexing technique of the B-ISDN, provides a connection-oriented service, a mismatch exists between the required connectionless service and the provided service. Therefore, additional protocols need to be used on top of ATM to provide also the required connectionless service.

In Chapter 3, an architectural framework has been presented, which places the protocols to be used in perspective. Two possible network architectures have been identified. The first one employs the so-called indirect method, where end-systems are connected by means of end-to-end ATM connections. The second one employs the so-called direct method, where end-systems are connected to CLSs in

the B-ISDN, which route packets through a connectionless overlay network to the proper destination.

The implementation of a CLS is one of the most difficult tasks in the design of the connectionless service. It has to support very high throughputs, while performing relatively complicated functions on the packets to be routed. In Chapter 4, candidate implementation architectures for a CLS have been identified. It has been shown that using some of these architectures, it is feasible to implement a highly-available and scalable CLS integrated within an ATM switch.

Performance modelling in Chapter 5 has revealed that CLSs, in which traffic shaping is performed in the module where the arriving traffic is buffered upon arrival, impose a much higher delay on the traffic than CLSs, in which buffering and traffic shaping are implemented in separate modules. Furthermore, it has been shown that the streaming mode of operation, where cells are processed on the fly, does not always lead to a lower delay than the message mode, where the cells of a packet are reassembled before processing. In particular, if the cells of a packet arrive on a connection with a high bandwidth assigned to it, the message mode of operation outperforms the streaming mode. Detailed analysis of the loss behaviour of the reassembly buffer, in Chapter 6, confirms the supposition that it is better to have few incoming connections with a high bandwidth than a large number of incoming connections with only little bandwidth.

Essential in the design of a connectionless service based on ATM is the connection management function, which instructs the signalling system to establish and release connections as needed for the transfer of packets. OCDR has been proposed as a mechanism for this function, in Chapter 7. Performance analysis has shown that it can lead to bandwidth, and hence cost reductions of up to 95%, compared to a permanent connection mechanism, without affecting the average delay experienced by packets.

Chapters 5, 6, and 7, were devoted to the development of performance models and their analysis. These models have been used to evaluate different design alternatives, and have lead to some of the conclusions presented above. On the other hand, the models have a value in itself, in that they can be used by designers and implementers to validate their design, and to determine optimal system parameters given the specific load on the system. As such, they can be considered as tailored tools, which can be used in the design of a system for providing connectionless communications using ATM.

The following tools have been developed in this thesis:

- a tool to assess the average delay of a CLS as a function of the used implementation architecture, the mode of operation, and the load (Chapter 5);
- a tool to assess the packet loss probability and occupancy distribution of a reassembly buffer as a function of its size and the load (Chapter 6); and
- a tool to assess, among others, the delay, reserved bandwidth, and signalling load of the OCDR mechanism as a function of its control parameters, and the load (Chapter 7).

8.2 Directions for Further Research

Connection management has been identified as an important function in a CLS or end-system. The OCDR mechanism has been proposed for the control of end-to-end connections, in case of the indirect method, and for the control of the connections between CLSs and end-systems. Other mechanisms for the control of connections between CLSs, on which traffic from numerous application processes will be multiplexed, should be further investigated. The VBC mechanism, which has been shown to be promising, needs to be analysed thoroughly.

In our analyses, we have considered the performance of a single node. An interesting direction for further research is the performance analysis of a network of nodes. Moreover, it is interesting to develop a tool that can be used to dimension an entire connectionless overlay network on top of ATM. Such a tool should support decisions regarding the switches in the ATM network, to which a CLS must be attached, the connections to be maintained between CLSs, and the bandwidth to be reserved on these connections.

The current ITU specifications of ATM have been used as a starting-point for our work. ATM, as it is defined now, is not the most suitable underlying communication system one can think of when designing a system for the provision of a connectionless service. It is a compromise between the demands of different types of applications and the opportunities provided by the current technology. However, since both application demands and technology are constantly changing, it may be necessary to modify or extend parts of ATM. In this process of reconsideration, it is also worthwhile to consider what modifications would make ATM more suitable for supporting connectionless applications. In partic-

ular, it may be interesting to investigate the possibility of connections which can be used on a best-effort basis, i.e., without the need for bandwidth reservation and without QoS guarantees.

Bibliography

[1] M. Ajmone Marsan, G. Conte, G. Balbo, “A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems”, *ACM Transactions on Computer Systems*, Vol. 2, No. 2, May 1984, pp. 93 - 122.

[2] G.J. Armitage, K.M. Adams, “Packet Reassembly During Cell Loss”, *IEEE Network*, Vol. 7, No. 5, September 1993, pp. 26 - 34.

[3] R. Atkinson, “ Default IP MTU for use over ATM AAL5”, RFC 1626, May 1994.

[4] ATM Forum, “ATM User-Network Interface Specification- Version 3.0”, ISBN 0-13-225863-3, PTR Prentice Hall, Englewood Cliffs, NJ, USA, September 1993.

[5] J.J. Barrett, E.F. Wunderlich, “LAN Interconnect Using X.25 Network Services”, *IEEE Network*, Vol. 5, No. 5, September 1991, pp. 12 - 16.

[6] Bellcore TA-TSY-000772, “Generic System Requirements in Support of Switched Multi-megabit Data Service”, Bellcore, Issue 3, October 1989, Supplement 1, December 1990.

[7] C. Blondia, O. Casals, “Statistical Multiplexing of VBR Sources: A Matrix-Analytic Approach”, *Proceedings of the workshop on Performance Aspects of ATM*, PTT Research, Leidschendam, The Netherlands, October 1991.

[8] G. Boiocchi, P. Crocetti, L. Fratta, M. Gerla, M.A. Marsiglia, “ATM Connectionless Server: Performance Evaluation”, *Proceedings IFIP WG6.4 International Workshop on Performance of Communication Systems*, Martinique, French Caribbean Island, January 1993, pp. 185 - 195.

[9] G. Boiocchi, P. Crocett, L. Fratta, M. Gerla, “Performance Evaluation of a Connectionless Multicast Service”, Elsevier, Teletraffic Science and Engineering, Vol. 1, *Proceedings of the 14th International Teletraffic Congress - ITC 14*, Antibes Juan-les-Pins, France, June 6 - 10, 1994, pp. 1121 - 1130.

[10] C. Bompard, A. Iera, V. Trecordi, “Analysis of the optimality of a multi-level bandwidth-negotiation algorithm for FDDI interconnection across B-ISDN”, *Proceedings 17th Conference on Local Computer Networks*, Minneapolis, Minnesota, September 1992, pp. 387 - 395.

[11] J.-Y. Le Boudec, A. Meier, R. Oechsle, H.L. Truong, “Connectionless data service in an ATM-based customer premises network”, *Computer Networks and ISDN Systems*, Vol 26, No. 4, 1994, pp. 1409 - 1424.

[12] D.F. Box, D.P. Hong, T. Suda, “Architecture and Design of Connectionless Data Service for a Public ATM Network”, *Proceedings IEEE Infocom ‘93*, San Francisco, CA, USA, March 1993, pp. 722 - 731.

[13] U. Briem, “Performance Comparison of Resource Sharing Schemes in a Connectionless Server on Top of ATM”, Elsevier, Teletraffic Science and Engineering, Vol. 1, *Proceedings of the 14th International Teletraffic Congress - ITC 14*, Antibes Juan-les-Pins, France, June 6 - 10, 1994, pp. 1109 - 1120.

[14] P. Buchholz. J. Dunkel, B. Müller-Clostermann, M. Sczittnick, S. Zäske, “Quantitative Systemanalyse mit Markovschen Ketten”, B.G. Teubner Verlagsgesellschaft, ISBN 3-8154-2056-3, Stuttgart 1994.

[15] R. Cáceres, “Efficiency of ATM Networks in Transporting Wide-Area Data Traffic”, Computer Science Division, University of California, Berkeley, 1991.

[16] CCITT Recommendation I.121, “Integrated Services Digital Network (ISDN), General Structure and Service Capabilities - Broadband Aspects of ISDN”, International Telecommunication Union, April 1991.

[17] CCITT Recommendation I.130, “Integrated Services Digital Network (ISDN), General Structure and Service Capabilities - Method for the Characterization of Telecommunication Services supported by an ISDN and Network Capabilities of an ISDN”, *CCITT Blue Book*, Fascicle III.7, International Telecommunication Union, Geneva 1989.

[18] CCITT Recommendation I.321, “Integrated Services Digital Network (ISDN), Overall Network Aspects and Functions, ISDN User-Network Interfaces - B-ISDN Protocol Reference Model and its Application”, International Telecommunication Union, April 1991.

[19] CCITT Recommendation F.812, “Telematic, Data Transmission, ISDN Broadband, Universal Personal Telecommunications and Teleconference Services, Operations and Quality of Service - Broadband Connectionless Data Bearer Service”, International Telecommunication Union, August 1992.

[20] CCITT Study Group XVIII, “AAL Type 5, Draft Recommendation text for section 6 of I.363”, SWP XVIII/8-5, Geneva, January 1993.

[21] H.J. Chao, D. Gosal, D. Saha, S.K. Tripathi, “IP on ATM Local Area Networks”, *IEEE Communications Magazine*, Vol 32, No. 8, August 1994, pp. 52 - 59.

[22] G. Chiola, “A Graphical Petri Net Tool for Performance Analysis”, *Proceedings 3rd International Workshop on Modeling Techniques and Performance Evaluation*, Paris, France, March 1987, pp. 323 - 333.

[23] J.P. Coudreuse, “Les résaux temporels asynchrones: du transfert de données à l’image animée”, *L’Écho des Recherches*, No. 112, 1983, pp. 33 - 48.

[24] P.J. Courtois, “Decomposability - Queueing and Computer System Applications”, ACM Monograph Series, Academic Press, ISBN 0-12-293750-X, 1977.

[25] P. Crocetti, G. Gallassi, M. Gerla, “Bandwidth Advertising for MAN/ATM Connectionless Internetting”, *Proceedings IEEE Infocom ‘91*, Bal Harbour, FL, USA, April 1991, pp. 1145 - 1150.

[26] P. Crocetti, L. Fratta, M. Gerla, M.A. Marsiglia, D. Romanò, “Connectionless support service in ATM”, *Proceedings ICCC 1992*, Genova, Italy, pp. 191 - 196.

[27] P. Crocetti, L. Fratta, M. Gerla, M.A. Marsiglia, D. Romanò, “Interconnection of LAN/MANs through SMDS on top of an ATM network”, *Computer Communications*, Vol. 16, No. 2, February 1993, pp. 83 - 92.

[28] D. Deloddere, P. Reynders, P. Verbeeck, “Architecture and Implementation of a Connectionless Server for B-ISDN”, Alcatel Bell Telephone, incl. personal correspondence, 1991.

[29] T. Demaria, G. Fioretto, A. Forcina, T. Moro, R. Vaglio, “Connectionless service handling within B-ISDN”, *CSELT Technical reports*, Vol. 20, No. 2, April 1992, pp. 103 - 111.

[30] A. Devinatz, “Advanced Calculus”, Holt, Rinehart and Winston, New York, 1968.

[31] W.J. van Dieten, “Performance Evaluation of a Connectionless Protocol over ATM using Matrix Geometric Methods”, Report of ‘Vakgroepopdracht’, University of Twente, Department of Computer Science, February 1994.

[32] B.T. Doshi, S. Dravida, “Congestion Control for Bursty Data in High Speed Wide Area Packet Networks: In-Call Parameter Negotiations”, *Proceedings ITC Specialist Seminar 7*, Morristown, NJ, 1990.

[33] B.T. Doshi, S. Dravida, P. Harshavardhana, “Performance and Roles of Bandwidth and Buffer reservation Schemes in High Speed Networks”, Elsevier, Teletraffic Science and Engineering, Vol. 1, *Proceedings of the 14th International Teletraffic Congress - ITC 14*, Antibes Juan-les-Pins, France, June 6 - 10, 1994, pp. 23 - 34.

[34] M. El Zarki, N. Shroff, “Performance Analysis of Packet Loss Recovery Schemes in Interconnected LAN-WAN-LAN Networks”, *Proceedings 3rd IFIP WG6.4 Conference on High Speed Networking*, Berlin, Germany, March 1991, pp. 337 - 351.

[35] ETS 300 217-1, “Network Aspects (NA): Connectionless Broadband Data Service (CBDS) - Part 1: Overview”, European Telecommunications Standards Institute, Final Draft, September 1992.

[36] ETS 300 217-2, “Network Aspects (NA): Connectionless Broadband Data Service (CBDS) - Part 2: Basic bearer service definition”, European Telecommunications Standards Institute, Final Draft, September 1992.

[37] ETS 300 217-3, “Network Aspects (NA): Connectionless Broadband Data Service (CBDS) - Part 3: Definition of supplementary services”, European Telecommunications Standards Institute, Final Draft, September 1992.

[38] ETS 300 217-4, “Network Aspects (NA): Connectionless Broadband Data Service (CBDS) - Part 4: Address screening supplementary service”, European Telecommunications Standards Institute, Final Draft, September 1992.

[39] W. Fischer, E. Wallmeier, T. Worster, S.P. Davis, A. Hayter, “Data Communications Using ATM: Architectures, Protocols, and Resource Management”, *IEEE Communications Magazine*, Vol. 32, No. 8, August 1994, pp. 24 - 33.

[40] M. Gerla, T.-Y. Tai, J.A. Suruagy Monteiro, G. Gallassi, “Interconnecting LANs and MANs to ATM”, *Proceedings 16th Conference on Local Computer Networks*, Minneapolis, Minnesota, USA, October 1991, pp. 259 - 270.

[41] M. Gerla, T.-Y. Tai, G. Gallassi, “LAN/MAN Interconnection to ATM: A Simulation Study”, *IEEE Infocom ‘92*, Florence, Italy, May 1992, pp. 2270 - 2279.

[42] M. Gerla, T.-Y. Tai, G. Gallassi, “Internetting LAN’s and MAN’s to B-ISDN’s for Connectionless Traffic Support”, *IEEE Journal on Selected Areas in Communications*, Vol. 11, No. 8, October 1993, pp. 1145 - 1159.

[43] R. Gusella, “A Measurement Study of Diskless Workstation Traffic on an Ethernet”, *IEEE Transactions on Communications*, Vol. 38, No. 9, September 1990, pp. 1557 - 1568.

[44] R. Gusella, “Characterizing the Variability of Arrival Processes with Indexes of Dispersion”, *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 2, February 1991, pp. 203 - 211.

[45] J. Habermas, “Moralbewusstsein und Kommunikatives Handeln”, Suhrkamp Verlag, Frankfurt am Main, 1983.

[46] R. Händel, M.N. Huber, S. Schröder, “ATM Networks: Concepts, Protocols, Applications”, 2nd edition, ISBN 0-201-42274-3, Addison-Wesley, Wokingham, England, 1994.

[47] B.R.H.M. Haverkort, “Performability Modelling Tools, Evaluation Techniques, and Applications”, Ph.D.-Thesis, University of Twente, ISBN 90-9003915-5, Enschede, The Netherlands, 1991.

[48] B.R. Haverkort, A.P.A. van Moorsel, D.-J. Speelman, “Xmgm: Performance Modeling Using Matrix Geometric Techniques”, *Proceedings of the Second International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS ‘94)*, Durham, NC, USA, January 1994, pp. 152 - 157.

[49] G.J. Heijenk, “Signalling in the B-ISDN - A survey of the state-of-the-art”, Memoranda Informatica 90-35, ISSN 0923-1714, July 1990.

[50] G.J. Heijenk, “B-ISDN Connectionless Service Definition”, RACE 1022 TG1 contribution, University of Twente, UT_TIOS_R1022_10, October 1991.

[51] G.J. Heijenk, I.G. Niemegeers, “Variable Bandwidth Connections for a Connectionless Service on ATM - Performance Modelling and Evaluation”, *IFIP Transactions C-4, Proceedings of the IFIP TC6 Workshop on Broadband Communications*, Estoril, Portugal, January 20 - 22, 1992, Ed. A. Casaca, North-Holland, 1992, pp. 361 - 372.

[52] G.J. Heijenk, A.P.A. van Moorsel, I.G. Niemegeers, “Performance of a Connectionless Protocol over ATM”, *Proceedings of the International Workshop on Advanced Communications and Applications for High Speed Networks (IWACA ‘92)*, Munich, Germany, March 16 - 19, 1992, pp. 123 - 130.

[53] G.J. Heijenk, M.A. Jordense, I.G. Niemegeers, "Description of a Multi-Media Multi-User Service", *Proceedings of the Second International Conference on Computer Communications and Networks (IC3N)*, San Diego, June 1993, pp. 213 - 220.

[54] G.J. Heijenk, M. El Zarki, I.G. Niemegeers, “Customer Creation and Combination in Queueing Networks”, Memoranda Informatica 93-35, ISSN 0924-3755, August 1993.

[55] G.J. Heijenk, X. Hou, I.G. Niemegeers, “Communication Systems Supporting Multimedia Multi-user Applications”, *IEEE Network*, Vol. 8, No. 1, January 1994, pp. 34 - 44.

[56] G.J. Heijenk, M. El Zarki, I.G. Niemegeers, “Modelling Segmentation and Reassembly Processes in Communication Networks”, Elsevier, Teletraffic Science and Engineering, Vol. 1, *Proceedings of the 14th International Teletraffic Congress - ITC 14*, Antibes Juan-les-Pins, France, June 6 - 10, 1994, pp. 513 - 524.

[57] G.J. Heijenk, I.G. Niemegeers, “Modelling the Reassembly Buffer in a Connectionless Server”, *Second Workshop on Performance Modelling and Evaluation of ATM Networks*, IFIP TC6, Participants Proceedings, Bradford, U.K., July 4 - 7, 1994, pp. 11 / 1 - 11 / 12.

[58] G.J. Heijenk, I.G. Niemegeers, “Modelling the Reassembly Buffer in a Connectionless Server”, extended version, Memoranda Informatica 94-43, ISSN 0924-3755, July 1994.

[59] G.J. Heijenk, “Simulation of a Connectionless Server for B-ISDN”, Memoranda Informatica 95-06, ISSN 0924-3755, January 1995.

[60] J. Heinanen, “Multiprotocol Encapsulation over ATM Adaptation Layer 5”, RFC 1483, July 1993.

[61] D.P. Hong, B.J. Vickers, T. Suda, “Connectionless Data Server for Public ATM Networks: Design and Performance”, Department of Information and Computer Science, University of California, Irvine.

[62] X. Hou, I.G. Niemegeers, “Defining Network Performance Parameters for a B-ISDN Signalling Network”, *Proceedings ICC ’94*, New Orleans, USA, May 1994, pp. 753 - 757.

[63] IEEE 802.6, “Distributed Queue Duel Bus (DQDB) Subnetwork of a Metropolitan Area Network (MAN)”, The Institute of Electrical and Electronics Engineers, December 1990.

[64] S.-I. IIsaku, M. Ishikura, “ATM network architecture for supporting the connectionless service”, *IEEE Infocom ‘90*, San Francisco, June 1990, pp. 796 - 802.

[65] ISO 7498-1, “Information Technology - Open Systems Interconnection - Reference Model - Part 1: Basic Reference Model”, Draft International Standard, 1992.

[66] ISO 8473-1, “Information Technology - Protocol for providing the connectionless-mode network service”, International Standard, 1993.

[67] ISO 8807, “ Information processing systems - Open Systems Interconnection - LOTOS - A formal description technique based on the temporal ordering of observable behaviour”, International Standard, 1989.

[68] ITU-T Recommendation I.150, “Integrated Services Digital Network (ISDN), General Structure - B-ISDN Asynchronous Transfer Mode Functional Characteristics”, International Telecommunication Union, March 1993.

[69] ITU-T Recommendation I.211, “Integrated Services Digital Network (ISDN), Service Capabilities- B-ISDN Service Aspects”, International Telecommunication Union, March 1993.

[70] ITU-T Recommendation I.311, “Integrated Services Digital Network (ISDN), Overall Network Aspects and Functions - B-ISDN General Network Aspects”, International Telecommunication Union, March 1993.

[71] ITU-T Recommendation I.327, “Integrated Services Digital Network (ISDN), Overall Network Aspects and Functions - B-ISDN Functional Architecture”, International Telecommunication Union, March 1993.

[72] ITU-T Recommendation I.350, “Integrated Services Digital Network (ISDN), Overall Network Aspects and Functions - General Aspects of Quality of Service and Network Performance in Digital Networks, including ISDNs”, International Telecommunication Union, March 1993.

[73] ITU-T Recommendation I.361, “Integrated Services Digital Network (ISDN), Overall Network Aspects and Functions - B-ISDN ATM Layer Specification”, International Telecommunication Union, March 1993.

[74] ITU-T Recommendation I.362, “Integrated Services Digital Network (ISDN), Overall Network Aspects and Functions - B-ISDN ATM Adaptation Layer (AAL) Functional Description”, International Telecommunication Union, March 1993.

[75] ITU-T Recommendation I.363, “Integrated Services Digital Network (ISDN), Overall Network Aspects and Functions - B-ISDN ATM Adaptation Layer (AAL) Specification”, International Telecommunication Union, March 1993.

[76] ITU-T Draft Recommendation I.364, “Integrated Services Digital Network (ISDN), Overall Network Aspects and Functions - Support of Broadband Connectionless Data Service on B-ISDN”, International Telecommunication Union, March 1993.

[77] ITU-T Recommendation I.371, “Integrated Services Digital Network (ISDN), Overall Network Aspects and Functions - Traffic Control and Congestion Control in B-ISDN”, International Telecommunication Union, March 1993.

[78] ITU-T Recommendation I.413, “Integrated Services Digital Network (ISDN), ISDN User-Network Interfaces - B-ISDN User-Network Interface”, International Telecommunication Union, March 1993.

[79] ITU-T Draft Recommendation Q.2931, “Broadband Integrated Services Digital Network (B-ISDN), Digital Subscriber Signalling No.2 (DSS2), Layer 3 Specification for Basic Call / Connection Control”, International Telecommunication Union, December 1993.

[80] ITU-T Study Group 11, “Broadband Capability Set 2 Signalling Requirements”, Document 1 / 11-9C, International Telecommunication Union, December 1993.

[81] ITU-T Study Group 18, “Support of Broadband Connectionless Data Service on B-ISDN”, Draft Recommendation I.364 and additions, Documents COMXVIII\RAPP\R116 and R130, International Telecommunication Union, 1993.

[82] V. Jacobson, “Multimedia Conferencing on the Internet”, *SIGCOMM ‘94*, Tutorial 4, London, England, August 1994.

[83] R. Jain, S.A. Routhier, “Packet Trains - Measurements and a New Model for Computer Network Traffic”, *IEEE Journal on Selected Areas in Communications*, Vol. 4, No. 6, September 1986, pp. 986 - 995.

[84] R. Jain, “The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling”, John Wiley & Sons, ISBN 0-471-50336-3, 1991.

[85] T. Kawasaki, S. Kuroyanagi, R. Takechi, K. Hajikano, “A study on high speed data communication system using ATM”, *Proceedings Globecom ‘91*, Phoenix, Arizona, December 1991, pp. 2105 - 2109.

[86] K. Kerkhof, A. van Halderden, “AAL type 5 to support the broadband connectionless data bearer service”, *Proceedings First International Symposium on Interworking*, Bern, Switzerland, November 1992.

[87] A. Kershenbaum, P. Kermani, G.A. Grover, “MENTOR: An Algorithm for Mesh Topological Optimization and Routing”, *IEEE Transactions on Communications*, Vol. 39, No. 4, April 1991, pp. 503 - 513.

[88] A. Kershenbaum, “Telecommunications Network Design Algorithms”, McGraw-Hill, New York, 1993.

[89] L. Kleinrock, “Queueing Systems, Volume I: Theory”, John Wiley & Sons, New York 1976.

[90] K. Kvols, E. Vázques, “Performance Evaluation of a Buffered ATM Connectionless Server”, *Proceedings of the First International Conference on Local Area Network Interconnection*, Research Triangle Park, NC, USA, October 1993, pp. 343 - 354.

[91] T. Van Landegem, R. Peschi, “Managing a Connectionless Virtual Overlay Network on Top of an ATM Network”, *Proceedings ICC ‘ 91*, Denver, Colorado, USA, June 1991, pp. 988 - 992.

[92] M. Laubach, “Classical IP and ARP over ATM”, RFC 1577, January 1994.

[93] W.E. Leland, D.V. Wilson, “High Time-Resolution Measurement and Analysis of LAN Traffic: Implications for LAN Interconnection”, *Proceedings IEEE Infocom ‘91*, Bal Harbour, FL, USA, April 1991, pp. 1360 - 1366.

[94] W.E. Leland, M.S. Tagu, W. Willinger, D.V. Wilson, “On the Self-Similar Nature of Ethernet Traffic (Extended Version)”, *IEEE/ACM Transactions on Networking*, Vol. 2, No.1, February 1994, pp. 1 - 15.

[95] W.P. Lidinsky, “Data Communications Needs”, *IEEE Network*, Vol. 4, No. 2, March 1990, pp. 28 - 33.

[96] R.S. McKinney, T.H. Gordon, “ATM for Narrowband Services”, *IEEE Communications Magazine*, Vol. 32, No. 4, April 1994, pp. 64 - 72.

[97] The Mathworks, “MATLAB User’s Guide”, The Mathworks Inc., Natick, Mass., February 1993.

[98] L. Mongiovi, M. Farrell, V. Trecordi, “A Proposal for Interconnecting FDDI Networks through B_ISDN”, *Proceedings IEEE Infocom ‘91*, Bal Harbour, FL, USA, April 1991, pp. 1160 - 1167.

[99] R. Nelson, “Matrix Geometric Solutions in Markov Models: A Mathematical Tutorial”, IBM Research Report, RC 16777, 1991.

[100] M.F. Neuts, “Matrix Geometric Solutions in Stochastic Models: An Algorithmic Approch”, John Hopkins University Press, Baltimore, 1981.

[101] I.G. Niemegeers, G.J. Heijenk, “The Evolution of Signalling from ISDN to B-ISDN”, *Proceedings of AMERICAS TELECOM 92, Technical Symposium*, Acapulco, Mexico, April 6 - 11, 1992, pp. 19 - 24.

[102] I.G. Niemegeers, G.J. Heijenk, X. Hou, “Signalling and Control in B-ISDN”, *Proceedings of the 4th International Conference on Advances in Communication and Control (COMCON 4)*, Rhodes, Greece, June 1993, pp. 745 - 757.

[103] R.T. Olson, L.H. Landweber, “NoW (No Waiting) Virtual Channel Establishment in ATM-like Networks”, Computer Sciences Technical Report #1082, University of Wisconsin - Madison, February 1992.

[104] R.T. Olson, L.H. Landweber, “Design and Implementation of a Fast Virtual Channel Establishment Method for ATM Networks”, *Proceedings IEEE Infocom ‘93*, San Francisco, CA, USA, March 1993, pp. 617 -627.

[105] D.S. Omundsen, A.R. Kaye, S.A. Mahmoud, "A Pipelined, Multiprocessor Architecture for a Connectionless Server for Broadband ISDN", *IEEE/ACM Transactions on Networking*, Vol. 2, No. 2, April 1994, pp. 181 - 192.

[106] R.O. Onvural, "Asynchronous Transfer Mode Networks: Performance Issues", ISBN 0-89006-662-0, Artech House, Norwood, MA, USA, 1994.

[107] J.B. Postel, "Internet Protocol", RFC 791, September 1981.

[108] D. Potier, "New Users' Introduction to QNAP2", INRIA, 1984.

[109] A. Pras, I.G. Niemegeers, "An Architecture for a distributed IS-PABX", *Proceedings International Conference on Local Communication Systems: LAN and PBX*, Palma, Spain, June 1991, pp. 13 - 26.

[110] M. De Prycker, "Asynchronous Transfer Mode: Solution for Broadband ISDN", 2nd edition, ISBN 0-13-178542-7, Ellis Horwood, New York, 1993.

[111] M. De Prycker, R. Peschi, T. Van Landegem, "B-ISDN and the OSI Protocol Reference Model", *IEEE Network*, Vol. 7, No. 2, March 1993, pp. 10 - 18.

[112] RACE 1022 Technology for ATD, "Report on Network Architecture for Connectionless Services", RACE Deliverable 22/TG01/XX/DS/P/019/B1, November 1991.

[113] RACE 1049 ATM Concept, "Outline specification for the provision of Connectionless Services for B-ISDN", RACE Deliverable 49/SIE/003/DS/B/b1, June 1991.

[114] RACE 2032 Combine, "Definitions of CL Server Interface Standard", RACE Deliverable D2/R2032/WP2/ /DEC92/SP, December 1992.

[115] RACE 2044 MAGIC, "User Perception and Network Performance", RACE Deliverable R2044/BTL/DP/DS/P/008/b1, March 1994.

[116] W. Rozenblad, B. Li, R. Peschi, "Interconnection of LANs/802.6 Customer Premises Equipments (CPEs) via SMDS on Top of ATM: a case description", *Proceedings IFIP TC6/WG6.4 Fourth International Conference on High Performance Networking*, IFIP Transactions C-14, Liège, Belgium, December 1992.

[117] W.H. Sanders, W.D. Obal II, M.A. Qureshi, F.K. Widjanarko, "UltraSAN Version 2: Architecture, Features, and Implementation", PMRL Technical Report 93-17, Department of Electrical and Computer Engineering, University of Arizona, October 1993, 22 p.

[118] H. Saran, S. Keshav, "An Empirical Evaluation of Virtual Circuit Holding Times in IP-over-ATM Networks", Department of Computer Science & Engineering, I.T.T. Delhi, India.

[119] A. Schmidt, R. Campbell, "Internet Protocol Traffic Anaysis with Applications for ATM Switch Design", *Computer Communication Review*, Vol. 23, No. 2, April 1993, pp. 39 - 52.

[120] W. Schödl, U. Briem, H. Kröner, T. Theimer, "Bandwidth allocation mechanisms for LAN / MAN interworking with an ATM network", *Computer Communications*, Vol. 16, No. 2, February 1993, pp. 93 - 99.

[121] J. Schot, L. Ferreira Pires, "Systematic Design of a Network Gateway using the FDT LOTOS", *Proceedings IEEE INFOCOM '91*, Bal Harbour, Florida, USA, March 1991.

[122] J. Schot, "The role of Architectural Sematics in the formal approach of Distributed Systems Design", Ph.D.-Thesis, University of Twente, ISBN 90-9004877-4, Enschede, The Netherlands, 1992.

[123] K. Shimokoshi, "Performance Comparison of Bandwidth Allocation Mechanisms for LAN / MAN Interworking through an ATM Network", *Proceedings Supercomm/ICC '94*, New Orleans, Louisiana, USA, May 1994, pp. 1405 - 1411.

[124] M. Sidi, W.-Z. Liu, I. Cidon, I. Gopal, "Congestion Control Through Input Rate Regulation", *IEEE Transactions on Communications*, Vol 41, No. 3, March 1993, pp. 471 - 477.

[125] SMDS Interest Group, "Protocol Interface Specification for Implementation of SMDS over an ATM-based Public UNI", SIG-TS-008 / 1994, Revision 1.0, May 1994.

[126] G.I. Stassinopoulos, I.S. Venieris, A. Bricca, R. Carli, "Bridged LAN Interconnection Through ATM", *Proceedings EFOC/LAN '91*, London, UK, June 1991, pp. 302 - 309.

[127] G. Stroebel, "Message Reassembly Times in a Packet Network", *IBM Journal on Research and Development*, Vol. 25, No. 6, November 1981, pp. 930 - 933.

[128] S.L. Sutherland, J. Burgin, "B-ISDN Interworking", *IEEE Communications Magazine*, Vol. 31, No. 8, August 1993, pp. 60 - 63.

[129] H. Suzuki, F. Tobagi, “Fast Bandwidth Reservation Scheme with Multi-link & Multi-path Routing in ATM Networks”, *Proceedings IEEE Infocom ‘92*, Florence, Italy, May 1992, pp. 2233 - 2240.

[130] T. Suzuki, “ATM Adaptation Layer Protocol”, *IEEE Communication Magazine*, Vol 32, No. 4, April 1994, pp. 80 - 83.

[131] P. Swart, “A Connectionless Service for Broadband ISDN”, M.Sc.-Thesis, University of Twente, Department of Computer Science, May 1992.

[132] J.S. Turner, “New Directions in Communications (or Which Way to the Information Age?)”, *IEEE Communications Magazine*, Vol. 24, No. 10, October 1986, pp. 8 - 15.

[133] J.S. Turner, “Design of an Integrated Services Packet Network”, *IEEE Journal on Selected Areas in Communications*, Vol. 4, No. 8, November 1986, pp. 1373 - 1380.

[134] I.S. Venieris, E.N. Protonotarios, G.I. Stassinopoulos, R. Carli, “Bridging remote connectionless LAN/MANs through connection oriented ATM networks”, *Computer Communications*, Vol 15, No. 7, September 1992.

[135] I.S. Venieris, J.D. Angelopoulos, G.I. Stassinopoulos, “Efficient Use of Protocol Stacks for LAN/MAN-ATM Interworking, *IEEE Journal on Selected Areas in Communications*, Vol. 11, No. 8, October 1993, pp. 1160 - 1171.

[136] B.J. Vickers, T. Suda, “Connectionless Service for Public ATM Networks”, *IEEE Communications Magazine*, Vol 32, No. 8, August 1994, pp. 34 - 42.

[137] B.J. Vickers, T. Suda, “Providing Connectionless Service in ATM Networks”, Department of Information and Computer Science, University of California, Irvine.

[138] C.A. Vissers, L. Ferreira Pires, J. van de Lagemaat, “Lotosphere, an attempt towards a Design Culture”, *Proceedings Third Lotosphere Workshop and Seminar*, Vol.1, 1992, pp. 1 - 30.

[139] C.A. Vissers, L. Ferreira Pires, D.A. Quartel, “The Design of Telematic Systems”, Lecture Notes, University of Twente, Enschede, The Netherlands, November 1993.

[140] R.J.F. de Vries, “Switch Architectures for the Asynchronous Transfer Mode”, Ph.D.-Thesis, University of Twente, the Netherlands, ISBN 90-72125-33-9, Enschede, 1992.

[141] M. Walch, A. Wolisz, R. Ruppelt, “Connection Management in a LAN to Broadband ISDN Gateway: A Simulation Study”, *Proceedings ICC ‘91*, Denver, Colorado, USA, June 1991, pp. 1671 - 1675.

[142] I. Widjaja, “Random Access for ATM LANs and WANs”, *Proceedings Supercomm/ICC ‘94*, New Orleans, Louisiana, USA, May 1994, pp. 39 - 43.

[143] W. Whitt, “The Queueing Network Analyzer”, *The Bell System Technical Journal*, Vol. 82, No. 9, November 1983, pp. 2779 - 2815.

[144] R.W. Wolff, “Stochastic Modeling and the Theory of Queues”, Prentice-Hall, ISBN 0-13-849845-8, 1989.

[145] A. Wolisz, M. Walch, R. Ruppelt, “Performance of Connection Management Policies for a LAN to Broadband ISDN Gateway”, *Proceedings International Conference on the Performance of Distributed Systems and Integrated Communication Networks*, Kyoto, Japan, September 1991.

[146] D.J. Wright, M. To, “Telecommunication Applications of the 1990s and their Transport Requirements”, *IEEE Network*, Vol. 4, No. 2, March 1990 , pp. 34 - 40.

[147] M. Yamamoto, T. Hirata, C. Ohto, H. Tode, H. Okada, Y. Tezuka, “Traffic Control Scheme for Interconnection of FDDI Networks through ATM Network”, *Proceedings IEEE Infocom ‘93*, San Francisco, CA, USA, March 1993, pp. 411 - 420.

[148] S. Yazid, H.T. Mouftah, T. Yang, “Fast Reservation Protocol and Statistical Multiplexing”, *Proceedings Supercomm/ICC ‘94*, New Orleans, Louisiana, USA, May 1994, pp. 733 - 737.

Abbreviations

AAL	-	ATM Adaptation Layer
AL	-	Alignment
ATM	-	Asynchronous Transfer Mode
AUU	-	ATM-user-to-ATM-user Indication
BASize	-	Buffer Allocation Size
BASTA	-	Bernoulli arrivals see time averages
BCDS	-	Broadband Connectionless Data Service
BCL	-	Broadband Connectionless
BCLS	-	Broadband Connectionless Service
B-ISDN	-	Broadband Integrated Services Digital Network
BOM	-	Beginning of Message
BTag	-	Begin Tag
CAC	-	Connection Admission Control
CATV	-	Cable Antenna Television
CBDS	-	Connectionless Broadband Data Service
CEP	-	Connection Endpoint
CIB	-	CRC Indication Bit
CLNAP	-	Connectionless Network Access Protocol
CLNIP	-	Connectionless Network Interface Protocol
CLNP	-	Connectionless Network Protocol
CLP	-	Cell Loss Priority
CLS	-	Connectionless Server
CLSM	-	Connectionless Service Module
CO	-	Connection-oriented
COM	-	Continuation of Message
CPCS	-	Common Part of the Convergence Sublayer
CPI	-	Common Part Indicator
CpP	-	Connection per Packet
CRC	-	Cyclic Redundancy Check
CS	-	Convergence Sublayer
CTMC	-	Continuous-Time Markov Chain
CUG	-	Closed User Group
D-MAP	-	Discrete-time Markovian Arrival Process

DTMC	-	Discrete-Time Markov Chain
DQDB	-	Distributed Queue Dual Bus
EOM	-	End of Message
ETag	-	End Tag
ETSI	-	European Telecommunications Standards Institute
FIFO	-	First In First Out
GFC	-	Generic Flow Control
HDTV	-	High Definition Television
HEC	-	Header Error Control
HEL	-	Header Extension Length
HLPI	-	Higher Layer Protocol Indicator
ICIP	-	Inter-exchange Carrier Interface Protocol
IDU	-	Interface Data Unit
i.i.d	-	independent identically distributed
IP	-	Internet Protocol
IPP	-	Interrupted Poisson Process
IS	-	Interconnection Structure
ISO	-	International Organization for Standardization
ITU	-	International Telecommunication Union
IWU	-	Interworking Unit
LAN	-	Local Area Network
LI	-	Length Indication
LOTOS	-	Language Of Temporal Ordering Specification
MAC	-	Medium Access Control
MAN	-	Metropolitan Area Network
Mbits	-	Megabits
MID	-	Multiplexing Identification
MMPP	-	Markov Modulated Poisson Process
NNI	-	Network Node Interface
NAP	-	Network Access Protocol
NP	-	Network Protocol
OCDR	-	On-demand Connection with Delayed Release
OSI-RM	-	Reference Model for Open Systems Interconnection
PAD	-	Padding
PASTA	-	Poisson arrivals see time averages
PC	-	Permanent Connection
PCI	-	Protocol Control Information
PDU	-	Protocol Data Unit
PL	-	Physical Layer

PPM	-	Packet Processing Module
PRM	-	Protocol Reference Model
PT	-	Payload Type
QNA	-	Queueing Network Analyzer
QoS	-	Quality of Service
RACE	-	Research and development in Advanced Communications technologies for Europe
RM	-	Routing Module
RPC	-	Remote Procedure Call
s	-	second
SAP	-	Service Access Point
SAR	-	Segmentation and Reassembly (sublayer)
SDU	-	Service Data Unit
SIP	-	SMDS Interface Protocol
SMDS	-	Switched Multi-megabit Data Service
SN	-	Sequence Number
SSCS	-	Service Specific Part of the Convergence Sublayer
SSM	-	Single Segment Message
ST	-	Sequence Type
TCP	-	Transport Control Protocol
TV	-	Television
UNI	-	User-Network Interface
UPC	-	Usage Parameter Control
UU	-	User-to-User indication
VBC	-	Variable Bandwidth Connection
VC	-	Virtual Channel
VP	-	Virtual Path
VCC	-	Virtual Channel Connection
VCI	-	Virtual Channel Identifier
VCL	-	Virtual Channel Link
VPC	-	Virtual Path Connection
VPI	-	Virtual Path Identifier
VPL	-	Virtual Path Link
VPN	-	Virtual Private Network
WAN	-	Wide Area Network

Index

Samenvatting

In het toekomstige openbare telecommunicatie netwerk, Breedband ISDN, kunnen toepassingen verwacht worden die gegevens willen kunnen versturen zonder eerst expliciet een verbinding op te zetten. Deze toepassingen verlangen een zogenaamde *connectionless service* van het netwerk. In ATM, de techniek die in het Breedband ISDN gebruikt wordt voor het samenvoegen en schakelen van gegevensstromen, moet echter altijd eerst een verbinding opgezet worden voordat communicatie kan plaatsvinden. Bovendien vindt in ATM transport van gegevens plaats in zogenaamde cellen met een vaste, kleine grootte, terwijl de toepassingen pakketten met gegevens van iedere willekeurige lengte moeten kunnen versturen. Daarom moet het Breedband ISDN uitgebreid worden met extra functionaliteit voor het bieden van een *connectionless service*. Het ontwerp en de analyse van zo'n *service* is het onderwerp van dit proefschrift.

Om de benodigde protocollen in perpectief te kunnen plaatsen is een architectureel kader ontwikkeld. De ontleding van de gewenste *connectionless service* in protocol entiteiten en de onderliggende ATM *service* resulteert in twee mogelijke netwerk architecturen. Volgens de eerste architectuur zijn eindsystemen van het Breedband ISDN onderling verbonden door ATM verbindingen. Volgens de tweede architectuur worden de eindsystemen door ATM verbindingen verbonden met speciale entiteiten, genaamd *Connectionless Servers*, die in het Breedband ISDN worden aangebracht. De *Connectionless Servers* zijn op hun beurt weer onderling verbonden door ATM verbindingen.

Voor de implementatie van een *Connectionless Server* wordt een aantal architecturen voorgesteld, die verschillen in de verdeling van de uit te voeren functies over modules. Ze worden vergeleken met betrekking tot effectiviteit, beschikbaarheid, schaalbaarheid, en in het bijzonder met betrekking tot de prestaties. Een *Connectionless Server* kan op twee manieren werken. In de zogenaamde *message mode* wordt gewacht tot een kompleet pakket is binnengekomen, alvorens de benodigde bewerkingen worden uitgevoerd, en het wordt doorgestuurd. In de zogenaamde *streaming mode* wordt met de bewerkingen begonnen zodra de ATM cel, die het eerste deel van het pakket bevat, binnen is. Wanneer de bewerkingen

zijn uitgevoerd wordt het deel van het pakket dat al binnen is meteen doorgestuurd. Status informatie wordt bewaard, zodat de cellen die de rest van het pakket bevatten ook kunnen worden doorgestuurd zodra ze aankomen.

Ten behoeve van de analyse van de prestaties van het ontwerp wordt in dit proefschrift een aantal modellen ontwikkeld. Een benaderend model van een *Connectionless Server* wordt gebruikt om de gemiddelde vertraging van een ATM cel te kunnen bepalen. Met behulp van dit model worden de verschillende implementie architecturen en de verschillende modes vergeleken. Het blijkt dat de *message mode* te verkiezen is boven de *streaming mode* wanneer de capaciteit van de ATM verbindingen tussen *Connectionless Servers* relatief hoog is. Is dit niet het geval, dan geeft de *streaming mode* de kleinste vertraging.

Voor de dimensionering van het deel van een *Connectionless Server* dat de pakketten reconstrueert uit de binnenkomende cellen, de zogenaamde *reassembly buffer*, wordt een ander, meer gedetailleerd model ontwikkeld en geanalyseerd. Dit model maakt het mogelijk de kans te bepalen dat een pakket verloren gaat omdat het aantal cellen dat bewaard moet worden de grootte van de buffer overschrijdt.

Om een *connectionless service* te kunnen bieden met het Breedband ISDN is een goed beheer van ATM verbindingen essentieel. Deze beheerfunctie zorgt ervoor dat de benodigde verbindingen voor het versturen van pakketten aanwezig zijn. In dit proefschrift wordt een nieuw mechanisme voorgesteld, dat gebruikt maakt van de verwachte correlatie tussen de aankomsttijden van pakketten om de capaciteit die van het ATM netwerk wordt gevraagd te minimaliseren. Met behulp van een model worden de optimale besturingsparameters van het mechanisme bepaald, en wordt het gedrag geëvalueerd. Ten opzichte van conventionele mechanismen kan, gegeven realistische veronderstellingen aangaande het te verwerken verkeer, de benodigde capaciteit met 95% verlaagd worden zonder de gemiddelde vertraging van pakketten te beïnvloeden.