

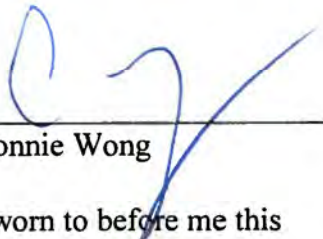


TRANSPERFECT

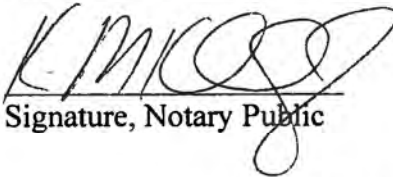
- ALBANY
- AMSTERDAM
- ATLANTA
- AUSTIN
- BARCELONA
- BERLIN
- BOSTON
- BRUSSELS
- CHARLOTTI
- CHICAGO
- DALLAS
- DENVER
- DUBAI
- DUBLIN
- FRANKFURT
- GENEVA
- HONG KONG
- HOUSTON
- IRVINE
- LONDON
- LOS ANGELES
- MIAMI
- MINNEAPOLIS
- MONTREAL
- MUNICH
- NEW YORK
- PARIS
- PHILADELPHIA
- PHOENIX
- PORTLAND
- RESEARCH TRIANGLE PARK
- SAN DIEGO
- SAN FRANCISCO
- SAN JOSE
- SEATTLE
- SINGAPORE
- STOCKHOLM
- SYDNEY
- TOKYO
- TORONTO
- VANCOUVER
- WASHINGTON, DC

City of New York, State of New York, County of New York

I, Connie Wong, hereby certify that the following is, to the best of my knowledge and belief, within the given parameters, a true and accurate translation, of the following document "Diplomarbeit Hettich" from German into English.

  
 \_\_\_\_\_  
 Connie Wong

Sworn to before me this  
Thursday, December 08, 2011

  
 \_\_\_\_\_  
 Signature, Notary Public

**KEVIN M KELLEY JR**  
 Notary Public - State of New York  
 No. 01-KE-6229268  
 Qualified in Queens County  
 Commission Expires October 4, 2014  
 \_\_\_\_\_  
 Stamp, Notary Public

Department of Communication Networks  
Aachen Technical University (RWTH)  
Prof. Dr.-Ing. B. Walke

Diploma Paper

# Development and performance evaluation of a Selective Repeat-Automatic Repeat Request (SR-ARQ) protocol for transparent, mobile ATM Access

by

Andreas Hettich

Student ID No. 181475

Aachen, April 17, 1996

Mentors:

Prof. Dr.-Ing. B. Walke, Associate Professor

Dipl.-Ing. D. Petras

This paper is for internal use only. All copyrights reserved by the mentoring department. We give no warranty for its content. Reproduction and publication of any kind are subject to the Department's permission.

I hereby confirm that I have conducted this work independently without the assistance from third parties – except for the official mentoring by the department. All literature used for this paper is listed completely in the Bibliography section.

Aachen, April 17, 1996

(Andreas Hettich)

# TABLE OF CONTENTS

---

<b>1</b>	<b>Introduction</b>	<b>4</b>
<b>2</b>	<b>Broadband ISDN (B-ISDN)</b>	<b>6</b>
2.1	Services in broadband ISDN.....	6
2.2	Switching in B-ISDN: <i>Asynchronous Transfer Mode (ATM)</i> .....	6
2.2.1	Structure of an ATM cell.....	7
2.2.2	ATM switching .....	8
2.2.3	ATM reference model.....	9
2.2.4	ATM service classes .....	10
<b>3</b>	<b>Mobile broadband system</b>	<b>11</b>
3.1	Overview.....	11
3.2	Bit transfer layer in an MBS .....	14
3.3	Cell structure of an MBS .....	15
<b>4</b>	<b>Logical link control (LLC) in an MBS</b>	<b>17</b>
4.1	Tasks and requirements.....	17
4.1.1	End-to-end error correction in an AAL .....	17
4.2	Requirements for an ARQ protocol.....	19
4.3	LLC and MAC layer interaction.....	21
4.4	Structure of the MAC layer.....	22
4.5	Structure of the LLC layer .....	22
<b>5</b>	<b>Link access protocols</b>	<b>25</b>
5.1	Conventional ARQ protocols.....	25
5.1.1	Stop-and-wait ARQ .....	26
5.1.2	Go back <i>n</i> (continuous) ARQ .....	27
5.1.3	Selective Repeat (SR) ARQ.....	28
5.2	Adaptive Selective Repeat (ASR) ARQ protocol .....	29
5.2.1	Elements of ARQ in the ISO 8802-2 standard.....	30

5.2.2	The <i>Ignore timer</i> .....	31
5.2.3	Treatment of time-critical ATM cells .....	34
5.2.4	The <i>Delay PDU</i> .....	34
5.2.5	Parallel ARQ instances .....	36
5.2.6	Acknowledge methods.....	37
<b>6</b>	<b>Implementation of the ASR ARQ protocol</b>	<b>40</b>
6.1	The Connection Handler .....	42
6.2	Send_Data.....	43
6.2.1	Send_Data_Object .....	44
6.3	Receive_Data .....	47
6.3.1	Receive_Data_Object .....	48
<b>7</b>	<b>The SIMCO3++/MBS simulator</b>	<b>50</b>
7.1	Overview.....	50
7.2	The stat_APP application layer.....	52
7.3	Sources.....	54
7.3.1	Poisson source.....	54
7.3.2	LAN source .....	54
7.3.3	Video source .....	55
7.3.4	CBR source .....	56
7.4	The MAC_Test_LCC MAC layer .....	57
7.5	The .sim_defaults configuration file.....	58
7.6	The graphic debugger .....	59
<b>8</b>	<b>Simulation results</b>	<b>62</b>
8.1	Introduction.....	62
8.1.1	Measured variables and performance parameters.....	62
8.1.2	Statistical power.....	63
8.1.3	Simulation parameters of the ASR ARQ.....	64
8.2	Optimization of the <i>ARQ Timer Delay</i> and <i>Ack_Threshold</i> parameters.....	65
8.2.1	ARQ Timer Delay.....	65
8.2.2	Ack_Threshold.....	66
8.3	Examining the causes for cell delays .....	68
8.3.1	Influence of the <i>Ignore timer</i> .....	68

8.3.2 Influence of the error model .....	71
8.4 Examining the effort required for discarding cells .....	72
8.5 Separation of acknowledgments and user data .....	74
8.6 Asymmetric traffic .....	75
<b>9 Summary and outlook</b> .....	<b>79</b>
<b>A List of abbreviations</b> .....	<b>81</b>
<b>List of figures</b> .....	<b>84</b>
<b>List of tables</b> .....	<b>85</b>
<b>Bibliography</b> .....	<b>86</b>

## CHAPTER 1

---

### Introduction

The telecommunications industry has been able to increase its sales despite a recession in the past few years. In Germany, sales rose from DM 11 billion to DM 23 billion in the period from 1990 to 1995 [25]. Another significant increase in sales is expected with the fall of the telecommunications monopolies in Europe in 1998. This illustrates the growing importance of telecommunications networks and services.

Many additional new services in the field of telecommunications became possible due to digital telecommunications systems. *ISDN (Integrated Services Digital Network)* deserves mentioning in this context. ISDN eliminates the variety of different network-user interfaces and offers all voice, text, and data services via a uniform network interface. In addition to digital fixed networks, new digital mobile networks became established based on the GSM (*Global System for Mobile Communication*) and DCS 1800 (*Digital Cellular System 1800*) standards. In Germany, the D1 network of Deutsche Telekom and the D2 network by Mannesmann Mobilfunk are based on the GSM standard. The E network by E-Plus was build according to the DCS standard.

The fixed bandwidth assignment from 64 kbit/s in the ISDN network and 22.8 kbit/s (gross) in the GSM system is designed for narrowband voice transmission (cutoff frequency 4 kHz) and cannot provide the high variable data rates that future broadband services such as video communication, data transmission in computer networks, and multimedia will require in general. This is why the ATM transmission method (*Asynchronous Transfer Mode*) was developed that is used in broadband ISDN (B-ISDN) and can provide a variable bit rate of 622 Mbit/s and more.

Since the need for higher data rates is beginning to show in mobile telephone networks, development of the UMTS system (*Universal Mode Telecommunication System*) was launched. It provides data rates of up to 2 Mbit/s when service-specific protocols are used.

An alternative approach is pursued under the *MBS* project (*Mobile Broadband System*). A system is to be designed that comes as close as possible to the functionality of broadband ISDN in landline networks. The targeted maximum gross data rate is 155 Mbit/s.

These very high data rates for a mobile telephone system can only be achieved by using accordingly high carrier frequencies in the ranges of 17, 40, or 60 GHz. There are unutilized bandwidths in these ranges. New powerful signal processing components will have to be developed for this purpose. Silicon technology can be used for 17 and 40 GHz, while 60 GHz require the use of gallium arsenide which is much more expensive.

A high subscriber density can be reached by small cell sizes with radii from 100 to 200 m. Small cell sizes can be achieved by using high carrier frequencies because there is additional resonance damping of the air in this range. This damping is around

14 dB/km and can increase by another 10 dB/km when it rains. This high damping ensures that the frequencies can be reused at a close distance.

The data is transmitted service neutral, and only the QoS (quality of service) parameters as defined in the ATM standards and the defined ATM service classes are specified. CAC (Connection Admission Control) analyzes the QoS parameters and traffic characteristics prior to setting up the connection. A connection that has been set up has to be monitored for compliance with the traffic characteristics agreed during setup, otherwise the quality of other connections may be impaired.

A physical channel in the MBS can carry 34 Mbit/s as gross data rate. If a virtual channel has a data rate of more than 34 Mbit/s, it will have to be split up over multiple frequency channels (multi-link transfer).

The protocols of the link layer have to meet special requirements; new concepts have to be implemented. A mobile ATM terminal is to be integrated into a landline ATM network while retaining the end-to-end relationship of the ATM adaptation layer. Poor transmission quality on the radio interface represents a special problem because does not allow the bit error rate required by ATM without additional measures. This can be remedied using forward error correction (FEC) in combination with a protocol for repeated transmission of ATM cells (Automatic Repeat ReQuest (ARQ)).

It is the purpose of this paper to design a new link access protocol and to assess it using a simulation. The protocol is based on known ARQ protocols and is adjusted to the special requirements of MBS.

At the outset, the paper gives an overview of *broadband ISDN* (Chapter 2) and the *Mobile Broadband System* (Chapter 3). Chapter 4 presents the logic link control (LLC) layer designed for mobile transparent ATM access and describes the special requirements of MBS. After an introduction to conventional ARQ protocols, Chapter 5 outlines the ASR ARQ protocol developed and implemented in this paper. Chapter 6 describes the implementation of the protocol in the system simulator. Chapter 7 presents the SIMCO3++/MBS simulator used for simulative performance assessment. Chapter 8 evaluates the capabilities of the protocol elements based on simulations. Chapter 9 concludes this paper with a summary and outlook.



# Broadband ISDN (B-ISDN)

## 2.1 Services in broadband ISDN

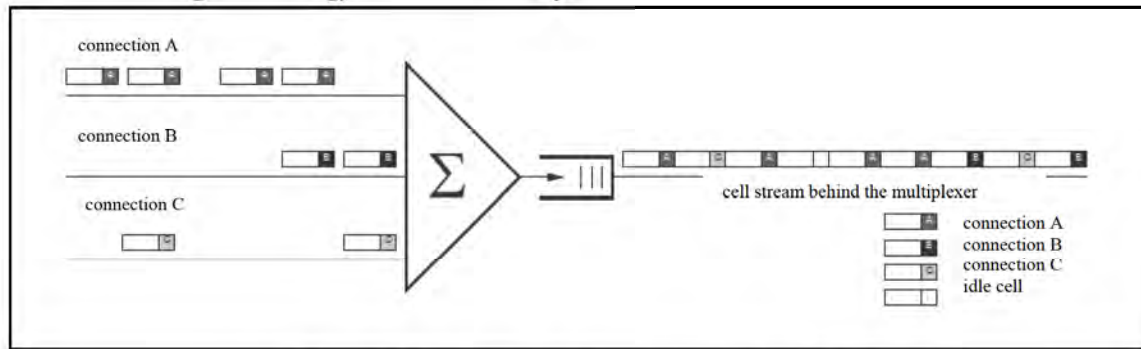
The integration of new broadband applications into the *Integrated Services Digital Network* (ISDN) requires data rates of up to 155 Mbit/s on the local loop. Some of these applications are services with a continuous data volume, others are cluster-type interactive services. The applications that generate continuous bit streams include voice transmission and video conferencing. Interactive services are characterized by varying bit rate requirements. A short query in a database, for example, can result in a reply with a very high data rate.

- Interactive services
  - telephony
  - image telephony
  - broadband video conference
- Retrieval services
  - access to databases
  - radio, TV, HDTV, video on demand
  - electronic newspaper
  - video mail
- Data communication
  - LAN connections
  - file transfer
  - CAM connections
  - high-resolution image transmission

The very diverse requirements of the broadband services can be met with difficulty only by using synchronous transmission methods (STDM). While over dimensioning the transmission capacity of synchronous channels reduces wait times, the capacity of the transmission medium is utilized poorly. The *asynchronous transfer mode* (ATM) can handle these requirements better.

## 2.2 Switching technology in B-ISDN: *Asynchronous transfer mode* (ATM)

The *asynchronous transfer mode* is the connection-related packet switching mode of B-ISDN. It combines the advantages of connection- and packet-oriented switching. The data streams to be transmitted are divided into blocks of a fixed length, the so-called ATM cells. The cells of various connections are transmitted in a time-interleaved manner via a physical channel. Different amounts of transmission capacity are assigned dynamically depending on the data rates of these connections. The cells are transmitted in the order of their arrival.

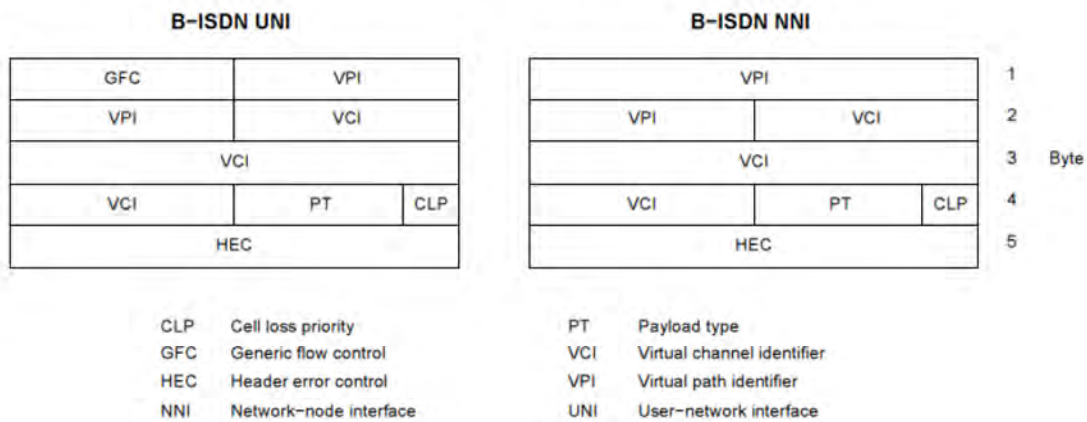


**Figure 2.1:** Statistical multiplex in the ATM method

The ATM multiplexer inserts “idle cells” into the joint data stream if none of the connections needs transmission capacity. This method is called statistical multiplexing and is particularly well suited for adapting to the dynamic communication behavior of very different connections.

**2.2.1 Structure of an ATM cell**

An ATM cell includes 53 bytes and is composed of a 5-byte header and a 48-byte information field that contains payload data. The connection of the cells is connection-oriented. All cells of one virtual connection take the same transmission path that is determined during setup by defining virtual channels. The network controls the cells based on the routing information stored in their headers. The header field contains the following parameters:



**Figure 2.2:** Header of an ATM cell

**VCI** *Virtual Channel Identifier*, 2 bytes

Identification of the virtual channel is used to distinguish among different simultaneous connections and the assignment of cells to these connections. The virtual channel identifier is assigned to one connection section only.

**VPI** *Virtual Path Identifier*, 8 or 12 bits

The VPI identifies a channel bundle. Multiple bundles can be distinguished that go in the same direction and each contain multiple virtual channels.

The switching network will be able to identify and forward cells from channels of the same bundle faster.

**PT** *Payload Type*, 3 bits

This identification of the type of information field for distinguishing payload and signaling information. The latter may contain information needed to update the routing tables that are kept in the switches. A switch has to analyze both the header and the payload data field of an ATM cell. If payload data is transmitted in the information field of an ATM cell, its contents is ignored.

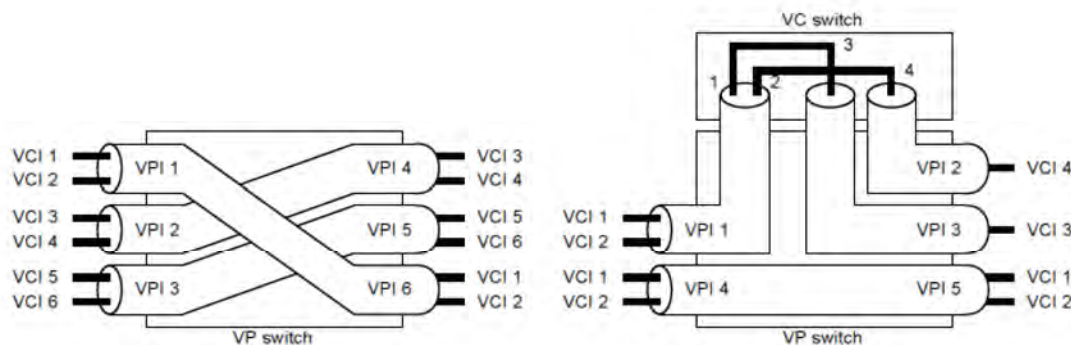
**HEC** *Header Error Control*, 1 byte

Since the header of an ATM cell contains data that is of particular significance for the transport of the cells, it is protected by a special check sequence. It is used for detecting and clearing errors.

**CLP** *Cell Loss Priority*, 1 bit

This parameter identifies low-priority cells that are discarded in the event of a queue overflow condition in the ATM switches.

## 2.2.2 ATM switching



**Figure 2.3:** Virtual path switching and virtual channel switching

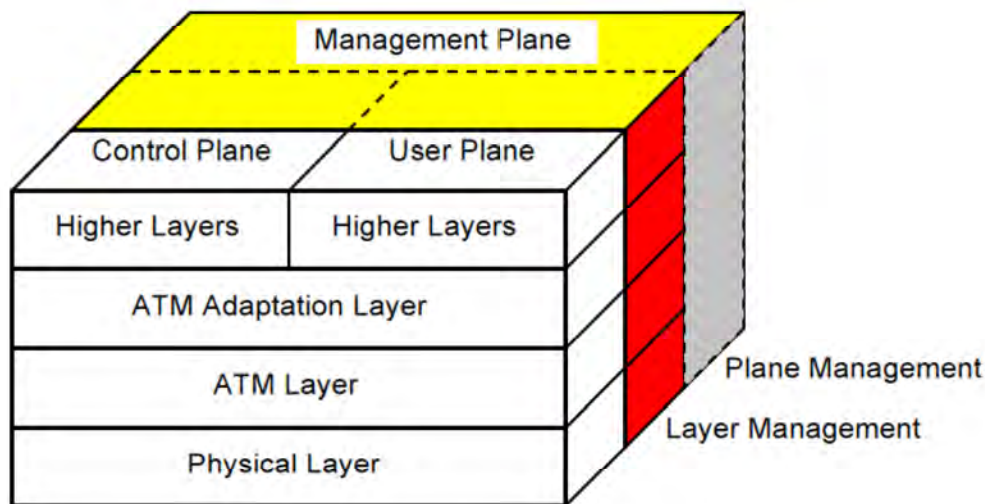
Like other packet switching methods, ATM cells are switched based on the routing information contained in their headers. Only the complete origin and destination addresses are sent during connection setup to keep this information as concise as possible for increased throughput. Other identifications of logical channels (VCI, VPI) are assigned for the various sections of the connection. The ATM switches establish routing tables based on the incoming control information. These tables contain an origin and destination identifier (line + virtual channel identifier). The cells are switched based on these tables as follows: The switch extracts the virtual channel identifiers from the incoming cells. It then enters the identifier of the next link based on the information contained in its routing table and routes the cell to the respective output of the switching network.

Figure 2.3 shows the switching elements used in the switches. There are VP and VC switches. A VP switch only analyzes the VPI values of the cells, which allows fast switching of the cells. The entries of the VCI fields remain unchanged. The VCI

identifiers are changed by VC switches only.

### 2.3.3 ATM reference model

Based on the recommendations of the OSI reference model, a four-layer reference model can be defined for ATM. These include the physical layer, the ATM layer, the ATM adaptation layer (AAL), and a layer that represents the functions of higher-order layers. Three different planes are also defined: the user plane, the control plane, and the management plane. The management plan includes the functions for plane management and the functions for layer management. The plane management is responsible for managing the entire system while the layer management controls each layer.



**Figure 2.4:** ATM reference model

The physical layer depends on the transmission medium and is to provide the functions for assembling and transmitting the cells via the physical medium. The tasks of the ATM layer include:

- Multiplexing and demultiplexing of cells from various connections
- Control of VCI- and VPI-oriented functions
- Generation and extraction of header information
- Generic flow control at the user-network interface (UNI) for setting up the connection.

The ATM adaptation layer (AAL) is used by the higher-order layers for adapting to the ATM layer. It performs the required segmenting of data streams and ensures a secure transmission. The AAL layer is divided into two sub layers:

1. The segmentation and reassembly (SAR) layer that maps the protocol data units of the higher-order layers to the ATM cells and vice versa.

2. The *Convergence Sub layer* (CS) that corrects undesired effects through cell transfer delay variations of different services.

For instance, the short data bursts generated during voice transfer are collected for a certain period of time in order to utilize the capacity of an ATM cell to the fullest. The receiver generates a continuous data stream again from the incoming ATM cells.

Transfer delay variations of the cells through the network are corrected by the ATM adaptation layer at the receiver. This is achieved by adding a constant time delay.

#### 2.2.4 ATM classes of service

In order to keep the number of the required protocols for the ATM layer at a minimum, the services are subdivided into four different classes according to the parameters “time reference between source and drain”, bit rate and connection type, and are shown in fig. 2.5.

	Class 1	Class 2	Class 3	Class 4
Time reference	Real time		Non-real time	
Bit rate	Constant	variable		
Connection type	Connection-oriented			Connectionless

Fig. 2.5: Classification of the services in ATM adaptation layer

Control data is also added to the content information depending on the service class used. The control data are used for restoring content information that is distributed into multiple cells. The control data is generated by the SAR sub layer during the distribution of the data in the cells. In the receiver, the respective SAR sub layer must join the data together again in the right sequence based on the control data.

1. In order to identify the individual cells, they are assigned sequence numbers. Based on this, it is possible for the receiver to detect any dropped cells. While the sequence number is present in all classes of service, additional back-up data and different segment types are only included in some classes. Due to the different proportions of control data, the resulting proportion of control data is 44-48 bytes [36].

## Mobile Broadband System

## 3.1 Overview

The RACE II research program (*Research and Technology Development in Advanced Communications Technologies in Europe*) advanced the development of third generation mobile radio systems from 1992 to 1995, which was supposed to enable the merger of systems such as GSM, DECT (*Digital European Cordless Telecommunications*) and trunked radio systems and their different areas of application, into a universal mobile radio system (*Universal Mobile Telecommunications System – UMTS*) with data rates of up to 2Mbit/s. At the same time, they strived to develop uniform end devices and expand the service to broadband services with high data rates [8].

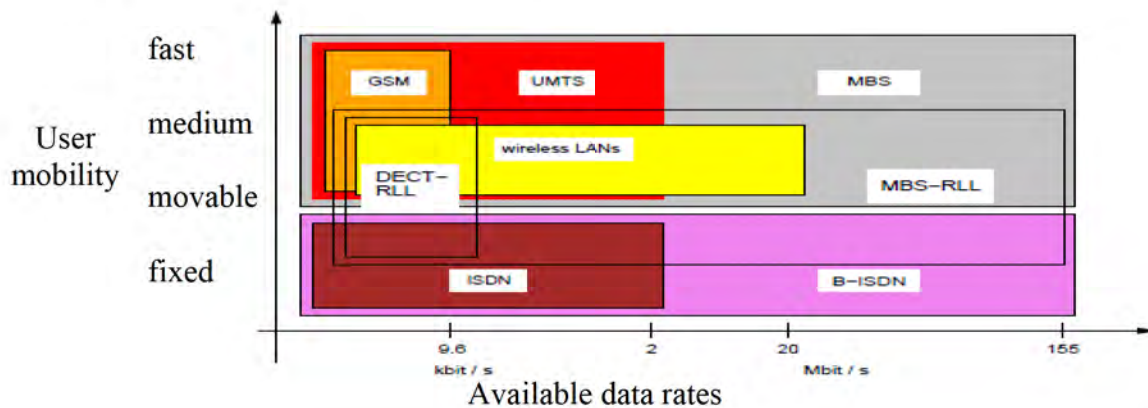


Fig. 3.1: MBS and other data networks

The MBS project that is being sponsored within the framework of RACE II deals with the connection of mobile users to stationary broadband networks (*Integrated Broadband Communication Network – IBCN*) at data rates of up to 155 Mbit/s. Apart from that, narrow band services still continue to be available. One of MBS' goals in particular is the provision of broadband ISDN services for mobile users and ATM transfer support.

Apart from the connection to broadband ISDN, MBS concept also includes possible collaboration with other systems such as UMTS. At the same time, the network type and the level of integration can vary

from a privately-operated MBS system with low service integration and mobility to a public MBS system with high integration, wide-range mobility and large coverage area [3]. In Fig. 3.1, MBS is linked with other data networks. It is evident that through MBS, the broad service spectrum of broadband ISDN is combined with the mobility of the mobile radio networks.

Due to the flexibility of MBS and the availability of B-ISDN services, a variety of different applications is possible. They are distinguished by the required data rates and the mobility of their users as shown graphically in Fig. 3.2.

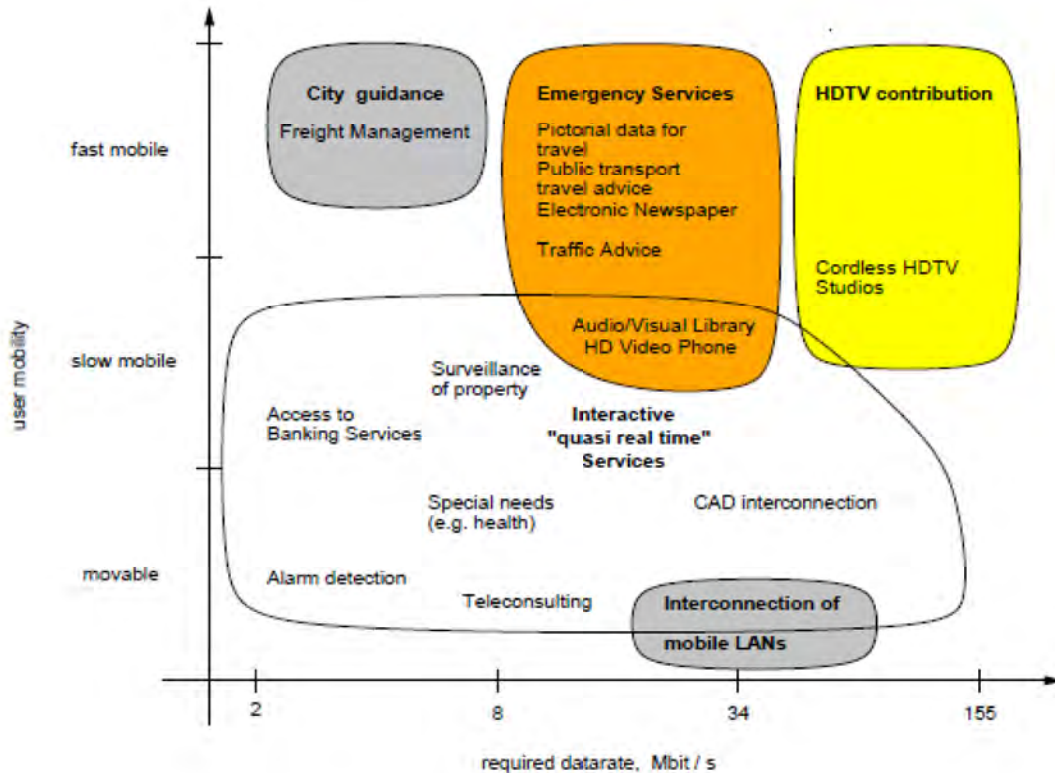


Fig. 3.2.: MBS applications and services

Specific conditions are required at the air interface during the integration of broadband services in mobile radio systems. They are a result of the mobility of the users and the utilization of a joint physical transfer medium. There are basically two different possibilities of implementation of mobile connection to fixed ATM network:

1. Provision of service-specific protocols for communication via air interface, whereby ATM transfer takes place only between the base station and the fixed network.
2. Transfer of ATM cells via the air interface whereby the mobile stations can transparently access the ATM network.

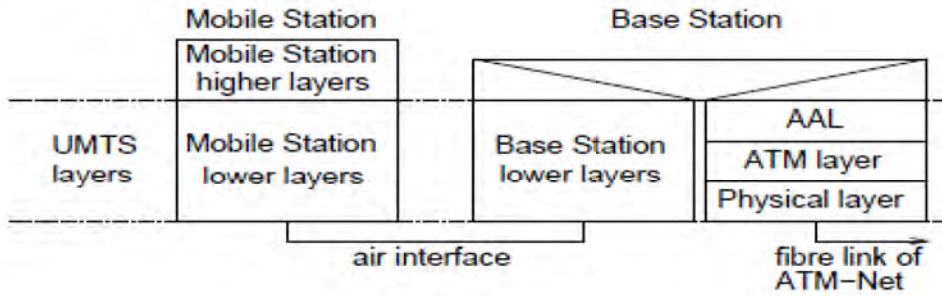
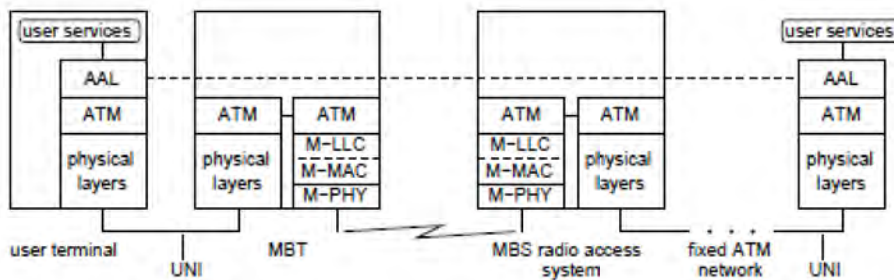


Fig. 3.3: Service-specific ATM transfer

The first process is used for example in UMTS and does not require any features for ATM transfer via the air interface. As a result, no channel access procedures that support the static multiplexes of the ATM are necessary and existing protocols can be adapted to the standards of different services. The disadvantages of this procedure are the need for different complex channel access procedures, lack of transparent ATM access and non-flexibility with regard to integration of new services (Fig. 3.3)



AAL	ATM Adaptation Layer	M-PHY	MBS Physical Layer
ATM	Asynchronous Transfer Mode Layer	MBT	Mobile Broadband Termination
M-LLC	MBS Logical Link Control Layer	UNI	User to Network Interface
M-MAC	MBS Medium Access Control Layer		

Fig. 3.4: Transparent, mobile ATM access

Transparent access means that the ATM adaptation layer must not be changed and can access the MBS-PHY layer like that of the ATM-PHY layer of a conventional ATM network. That is why the second approach only requires a service-independent MAC scheme (*Medium Access Control*) to transfer in asynchronous transfer mode. ATM services to be added in the future will also be supported by this universal solution. However, channel access procedures that support static multiplexes of the ATM are necessary. This procedure can provide the same functionality but not the same quality of service like in B-ISDN. This approach was selected in the MBS system (Fig. 3.4) [15].



### 3.2 Bit transfer layer in MBS

The bit transfer layer (Physical layer) represents the lowest layer in ISO-OSI reference model and specifies the protocols for transferring the bits to the physical channel. The transfer of high data rates that require large broadband is included in MBS. As a result, frequencies in the 60 GHz frequency range were selected as carriers. On the one hand, these frequencies are otherwise available, and on the other hand, the atmospheric oxygen attenuation is very high. In addition to the parameters just mentioned above, the radius of a radio cell depends mainly on the noise factor, the required signal to noise power ration in the receiver, the bandwidth and the channel characteristics. Taking these factors into account results in cell radii in the scale if  $R = 100$  to  $200m$ . Due to damping, the reusability intervals of a frequency are very short. This allows high user densities to be achieved. Cell planning as is required for operating second generation mobile communication networks can no longer be carried out due to the microcellular structure in MBS.

It requires the use of procedures for dynamic channel allocation.

For synchronization, the mobile station establishes the time interval on the downlink that it receives delayed around the signal propagation time. This leads in uplink to maximum asynchronizations between mobile stations near the base station and those at a large distance of  $T_{asynch} = 2.R/c \approx 0.666\mu s$  to  $1.33\mu s$  ( $c$  is the velocity of light) because the transmit signal of the mobile station arrives in the base station delayed at an equivalent run time. A measurement of the order  $T_{asynch}$  by the base station, which is also called loop delay, as occurs e.g. in GSM networks is not included in MBS.

Frequency range for the downlink	62 to 63 GHz
Frequency range for the uplink	65 to 66 GHz
Number of frequency channels	32
Bandwidth per frequency channel	30 MHz
Duplex splitting	2 FDD
Multiplex procedure	FDMA / TDMA (hybrid)
Slot length	21.33 $\mu s$
Guard time	1.33 $\mu s$
Modulation procedure	QPSK / 16 QAM
Data rate	20 M Symbols/s ( $T_{symbol} = 50 ns$ )
ATM cells / slot	1 (= 424 bits)
Asynchronization	$T_{synch} \approx 0.666$ to $1.33\mu s$ ( $\approx 13$ to $26$ symbols)
Radio cell radius	100 -200 $m$

Table 3.1: System parameter of MBS system

The system basically represents a hybrid FDMA / TDMA channel structure (FrequencyTime Division Multiple Access). The frequency range is subdivided into 32 channels for each 30 MHz bandwidth that are separated from each other by guard bands. The time range is split into time slots. The length of a slot is  $T_{slot} = (20 + 1.33) \mu s$ . It consists of a guard time  $T_{guard} = 1.33 \mu s$  on the uplink, which is supposed to prevent time overlapping of the packets that are sent on the same carrier in succession from different mobile stations. The length of the guard time is

equivalent to the maximum asynchronization  $T_{\text{asynch}}$  such that irrespective of the position of a mobile station within a cell, the sum of the delay and length of a packet does not exceed the length of the slot. The  $1.33\mu\text{s}$  correspond to the signal propagation of base to mobile station and back again, if the mobile station assumes largest distance possible from the base station. On the downlink, guard time is required to adjust the transmitter when using the power control method.

The allocation of a slot is shown in Fig. 3.5. In addition to the actual data ( $2 \times 168$  symbols), each slot contains a training sequence (15 symbols) in the center, which serves to assess the channel step response of the equalizer as well as synchronization. Three different slot lengths are shown in Fig. 3.5 but only normal and very short slots are deployed. Normal slots serve for transferring data while very short slots are used for transferring dynamic parameters and collision resolution.

If QPSK (Quaternary Phase Shift Keying) is used, a symbol is equal to two bits and in the case of the 16 QAM (Quadrature Amplitude Modification) a symbol would be equal to four bits. The cost as well as the error susceptibility of the second modulation procedure is clearly on a higher scale and as such, QPSK should be deployed in the first level of implementation.

QPSK is also taken as the modulation procedure so that  $168 \cdot 2 \cdot 2 \text{ bits} = 672 \text{ bits} = 84 \text{ bytes}$  is available in the case of a normal slot., out of which an ATM cell generally needs 53 bytes, and as overhead, MAC and LLC layer split up the remaining 53 bytes. A slot of very short length which will now be called subslot, has  $24 \cdot 2 \cdot 2 \text{ bits} = 96 \text{ bits} = 12 \text{ bytes}$  [15].

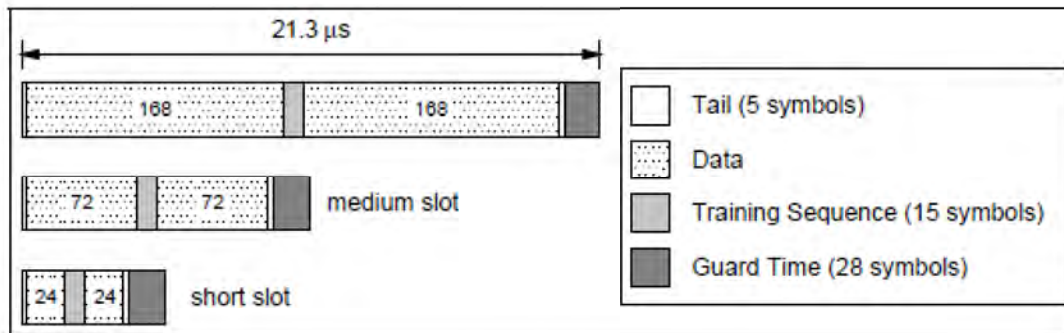


Fig. 3.5 Layout of a slot

### 3.3 MBS cell structure

The MBS cell structure is similar to that of GSM (see Fig. 3.6)

A Base Station Controller (BSC) has a link to the fixed ATM network to which other BSCs are connected. The link layer as well as the ATM layer is located inside the controller. The actual receiving / transmitting devices (Base Station Transceiver (BST) are located inside a Base Transceiver Station (BTS). The BTS are spatially distributed within the coverage area of a BSC and give the BST a local designation. Multiple *base station transceivers* can be installed on a *base transceiver station*. During the process, each *base station transceiver* transmits on another

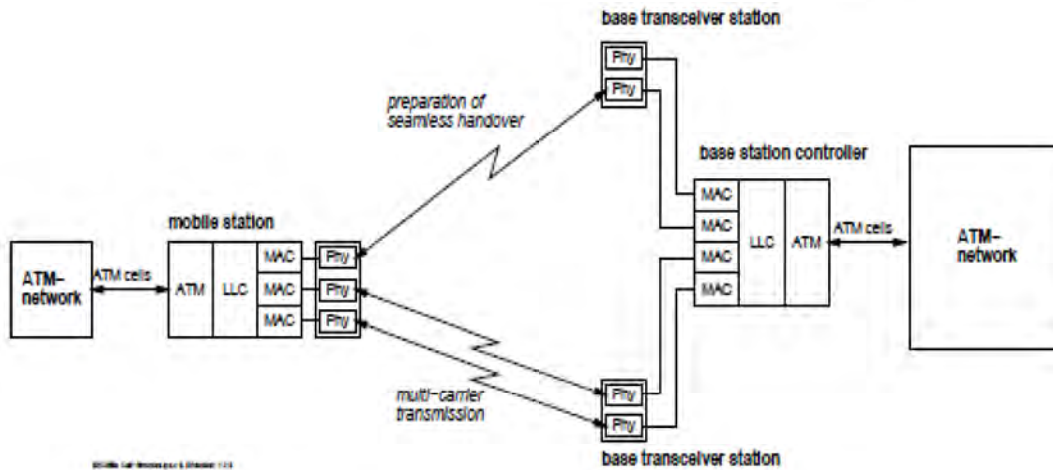


Fig. 3.6: Cell structure of the MNS system

another frequency. Only elements of the physical layer are present in *base transceiver stations* in contrast to GSM. A mobile station (MS) contains all MBS layers. On the other hand, a mobile network that is made up of several ATM end devices can also be connected to a mobile station.

In order to be able to cope with the high data rates of the connected end devices, the mobile station must set up multiple physical channels to the same or to another *base station controller* in certain circumstances (*Multilink transfer*). A physical channel can transfer 77,000 ATM cells = 34 Mbits including signaling. However, the mobile broadband system is supposed to be able to control data rates up to 155 Mbits, which is equivalent to five simultaneously operated frequency channels.

In the event of decreasing reception quality, the mobile station must be able to execute a *handover* to another *base station transceiver* of the same or another *base station controller*. This handover must occur seamlessly i.e. no data must be missing or delayed during *handover*. This is only possible if a second physical channel was set up for another *base transceiver station* on time and if both channels were operated simultaneously for a certain period of time. Since interferences can occur more or less without forewarning through fading, shading or the like in the 60 GHz range, signaling links to other BTS must be constantly available at the least.

### Logical Link Control (LLC) in MBS

The LLC layer is located above the MAC layer and together they form the link layer in the ISO / OSI reference model.

MBS is supposed to allow mobile, transparent access to ATM networks. Transparent in this case means that the protocols of the ATM adaptation layer remain unaffected by the MBS system. In MBS, data is supposed to be available in the same quantity and quality as in fixed ATM networks. The function of the LLC in this case is to allow quality standards that are nearly as good as those in fixed ATM networks.

#### 4.1 Functions and requirements

The ATM concept is based on the assumption that high transfer quality is guaranteed which certainly applies to the optical glass fiber technology currently in use. However, the transfer via air interface is much less reliable and depends highly on the surrounding conditions. Since however, glass fiber technology cannot fully prevent the occurrence of transfer errors, a simplified process for end to end error correction was executed within the ATM adaptation layer in ATM networks depending on service type. Apart from that, a header error control was carried out within the ATM layer. The explanation below shows that the required error performance cannot be achieved with this simplified error correction procedure if one or more glass fiber links are replaced with radio links. Instead, it requires additional error correction procedures that are specially tailored to the characteristics of radio transfer. Due to the required transparency for AAL protocols, as well as the reasons listed in chapter 4.1.1, the additional error correction procedures required can only be carried out directly at the air interface. As such, this presents the classic case of an LLC protocol on radio link.

##### 4.1.1 End to end error correction within AAL

Due to vast number of applications, end to end error correction procedures within different AAL types face different challenges.

In AAL type 1 and AAL type 2, a simple procedure for detecting missing or erroneously inserted ATM cells is used so that the bit error rate as well as the cell loss rate is transferred unaltered to the AAL service user. Alternatively, isolated bit errors can be detected and possibly corrected using FEC [19]. A very high bit error rate within the ATM layer due to a radio link can lead to the impairment of quality of service required by a specific service in such a way that the reproduction of a voice service can be altered unsatisfactorily. That is why in such configuration of an

ATM network, specific services cannot be supported without additional error correction procedures at the air interface within the ATM layer.

In AAL type 3 / 4 as well as AAL type 5, a typical transport protocol with automatic repeat request (ARQ) is executed within the service specific convergence layer (SSCS), which relies on the features for detecting cell losses and bit errors of the lower sub layers common part convergence sub layer (CPCS) and segmentation and reassembly (SAR) [12]. No SSCS protocol has been specified so far for AAL type 3 / 4. Service specific connection-oriented protocol (SSCOP) [20] was specified for AAL type 5. It runs with very long sequence numbers (24 Bits) and has information frame of a maximum length of 64 Kbytes. At an hypothetical end-to-end bit error rate of  $10^{-7}$ , 1 Kbyte frame yields frame loss rate of  $10^{-3}$  which allows ARQ protocol [3] to be executed effectively. However, typical bit error rates on FEC-only protected air interface is approximately  $10^{-3}$  to  $10^{-5}$  [7, 33]. This means that frame loss rate for frames in the 1 Kbyte range is above 8%, [which makes error correction by a transport protocol much more ineffective with regard to overhead and other delays due to repeat transfers, as error correction of individual cells within an LLC protocol at the air interface.]

In summary, it is clear that the algorithms within the individual AAL types are tailored to a specific error behavior of the ATM layer. At the same time, the required bit error rates and cell loss rates do not only depend on the AAL types but also on the specific application and its data that is being transferred via a virtual channel (VC). These requirements are indicated in the VC-specific parameters [18] which must be observed. Thus, the error correction procedures at the air interface must adjust the cost for error correction to the QOS standards of the individual VCs [27].

Two examples illustrate the required level of adaptation:

**Telephone service** uses AAL type 1 and is adapted to the CBR service class within the ATM layer. It is identified by a relatively low, constant data rate and has very high standards for the authenticity behavior, which is characterized by the fact that ATM cells with transfer delays that exceed a specific maximum value are rejected. Furthermore, phone service is relatively insensitive to transfer errors and as such, bit error rates of  $10^{-4}$  only lead to minimum loss in voice quality. As a result, the ARQ protocol can be set up relatively easy in the LLC layer at the air interface for CBR services, and can possibly only execute a single repeat transfer for each ATM cell every time [37]. It is rejected when multiple transfer errors of the same ATM cells occur.

**Non time-critical data service (e.g. file transfer)** is used by AAL type 3 / 4 or AAL type 5 and is adapted to the ABR service class within the ATM layer. Values for a maximum delay are not specified. An average delay with high value is specified if necessary. The bit error rate should be below  $10^{-9}$  and is reduced much more by the AAL. An HDLC-type [16] ARQ protocol can be executed in the LLC layer at the air interface for such services.

A compromise must be made between the residual bit error rate and transfer delays for real time-oriented services of AAL type 2 that are adapted

to the VBR class within the ATM layer, and the number depends on the repeat transfers according to the requirements of a VC (QoS parameter), also on current constraints such as volume of load within a station and also on the shared radio resources (static multiplexes).

Since the LLC protocol is located under the ATM layer, the different requirements of the AAL types with regard to error behavior of the ATM layer must be moved to the ATM classes. At the same time, the VCs of ATM type 1 are adapted to CBR for the most part, AAL type 2 to VBR and AAL type 3 / 4 as well as AAL type 5 to ABR.

#### 4.2 Requirements for ARQ protocol

The question now is whether in the LLC layer an ARQ protocol has to be specified for each ATM class or whether a generic protocol can be reused for all classes. This paper examines the characteristics of such a generic protocol that can be parameterized for each ATM class. Afterwards, we will examine what way this protocol can be simplified if it is to be tailored to the needs of a specific ATM class. The solution used in a real system will vary depending on the actual constraints such as bandwidth, user profile, and degree of professionalism or costs. One should be able to use all the analyses compiled in this paper as design guideline for this purpose.

If multiple VCs run parallel between two stations, ARQ protocol can be executed for individual VCs or for a series of multiplexed VCs (Fig. 4.1). This results in a structure of several independent ARQ instances that are each assigned a VC or multiple VCs that are bundled by a VC multiplexer. The VC multiplexer replaces the multiplex feature within the ATM layer. An ARQ instance transfers its ARQ frame via an ARQ channel assigned to it. Parallel ARQ channels (between the same partner instances) can be bundled by an ARQ multiplexer to a MAC channel. However, for specific ARQ channels, it is practical to also pass through directly to the MAC layer.

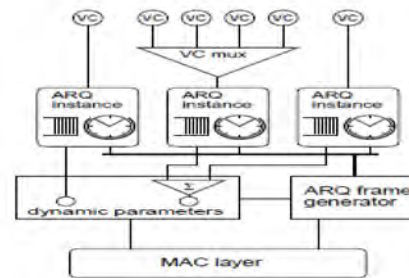


Fig. 4.1: Distribution of different VCs to ARQ instances

An important item that affects the complexity and hence the cost of the LLC layer, is the number of parallel ARQ instances that resulting from the process of assignment of VCs to ARQ instances. Since VCs have very different standards for cell loss rates or transfer

delays, it is practical to put together only VCs with similar requirements by a VC multiplexer above an ARQ instance.

Parallel ARQ instances have the advantage that ATM cells of time-critical VCs can pass those of non-critical VCs and do not have to wait first for their transfer. This reduces the transfer delays due to the idle time on a transfer time slot.

Furthermore, CBR-type VCs should not be multiplexed with other VCs but processed by its own ARQ instance, which allows the separate handover of its ARQ frames to the MAC layer.

For real-time oriented VBD-type VCs, an example mentioned in [27] is used to explain that the idle times in the resequencing buffers of the receiver can be reduced if the association of erroneously received and repeat transfer requested ARQ frames to specific VCs is known. In the event of the loss of an ARQ frame with an ATM cell, all other cells with higher sequence numbers in the resequencing buffer must be cached in order to wait for the repeat transfer of the lost cells. However, if the lost cells belong to a VC other than the waiting cells, then it is not necessary to delay the cells already successfully received. When an ARQ instance only transfers the ATM cells of a single VC, such unnecessary increase in transfer delay is impossible.

The idle times in the resequencing buffer can also be reduced in the case of multiple VCs within the same ARQ instance by the fact that fast transfer of positive or negative ACKs [acknowledgement] is possible. This is the case for example in high-rate symmetrical channels that cover a large part of the capacity of a carrier, hence ACKs can very often be transferred simultaneously to an information frame in the opposite direction after short idle time. Since transfer delays occur several times across the length of a time slot, its maximum range depends highly on the underlying bandwidth of the system and hence depends on a specification of a limit for identifying high-rate channels of the respective constraints. Fast transfer of ACKs can also be facilitated through special processes within the MAC layer where, if ATM cells are transferred in ARQ frames, a short time slot on the backward channel for transferring a feedback or an ACK is reserved automatically [29].

In summary, the following rules can be deduced from the assignment of VCs to ARQ instances:

- A multiplex above an ARQ instance is only practical for VCs with similar requirements for cell loss rate and transfer delays.
- All ABR-type VCs should be processed by the same ARQ instance.
- For time-critical CBR-type or VBR-type VCs, an ARQ instance must be installed separately.
- In the case of guaranteed, high-speed transfer of positive or negative ACKs, time-critical VCs can also be multiplexed above an ARQ instance.

### 4.3 Interaction between LLC Layer and MAC Layer

Section 5.2 will take a detailed look at the functionality of the protocol within an ARQ instance; before doing so, however, this section will explain the algorithms of the ARQ multiplexer, as well as how they interact with the algorithms of the MAC layer. To do so, the requirements for the MAC layer, as well as the functionality will be described in general terms. The services provided by them will then be explained, which can be used advantageously by the LLC layer.

The requirements for the transparent transmission of ATM cells via the wireless interface across multiple mobile stations and one central base station result in a channel access protocol, which expands the static multiplexes of the ATM multiplexers to the special scenario, which is characterized by competing multiple accessing of mobile stations, which are not easy to coordinate. During this process, the protocol must allocate the transmission media to the individual mobile stations based on the transmission requirements they have at that specific moment; although a TDMA structure where the time slots are designed for the transmission of individual ATM cells is used as the basis. Moreover, a full duplex transmission is required. Due to the competition and the related collisions, the uplink forms the critical component for the channel access protocol.

In theory, there is a series of protocols in which the base station – as the central instance – allocates the transmission capacity to the mobile stations based on the individual time slots [29, 1, 5]. During this process, a cell – composed of multiple mobile stations and one base station – forms an ATM multiplexer, which can be modeled as a shared queue system. Since the data transfer rate of the wireless interface (34 Mbit/s per carrier in MBS [33]) is relatively low compared to a fixed network, the base station must use a scheduler with static or dynamic priorities when assigning time slots for both the uplink and the downlink [14, 26, 22] in order for the ATM cells of the real-time-based VCs to be able to achieve the necessary low transmission delays [28]. The scheduler is located in the base station. The use of dynamic priorities will be assumed herein due to the special conditions of the wireless interface.

Since an ATM cell is embedded in an ARQ frame, the header of which contains both an acknowledgement, as well as the current dynamic parameters, it would not make sense to temporarily store the ATM cells to be sent within the MAC layer while waiting for a transmission time slot. Instead, an ATM cell should not be embedded in an ARQ frame and delivered to the MAC layer until directly before its transmission by the *ARQ frame generator*.

The scheduler of the MAC layer determines a mobile station to which, or from which, an ATM cell can be transmitted for each time slot and does so based on the dynamic parameters. The LLC layer must then generate an ARQ frame, which not only contains an ATM cell, but whose header contains an acknowledgement, as well as a set of dynamic parameters in case of an uplink. The ATM cells with the highest *transmission urgency* are then sent. This urgency corresponds to the priority in the scheduler. One possible processing strategy is G/D/1/FCFS/RU-NONPRE (relative urgency) [38], in which the ATM cells are prioritized for transmission based on their wait time and the connection-specific service quality requirements. This allows the fluctuation in the delay (delay jitter) of each ATM cell, as well as

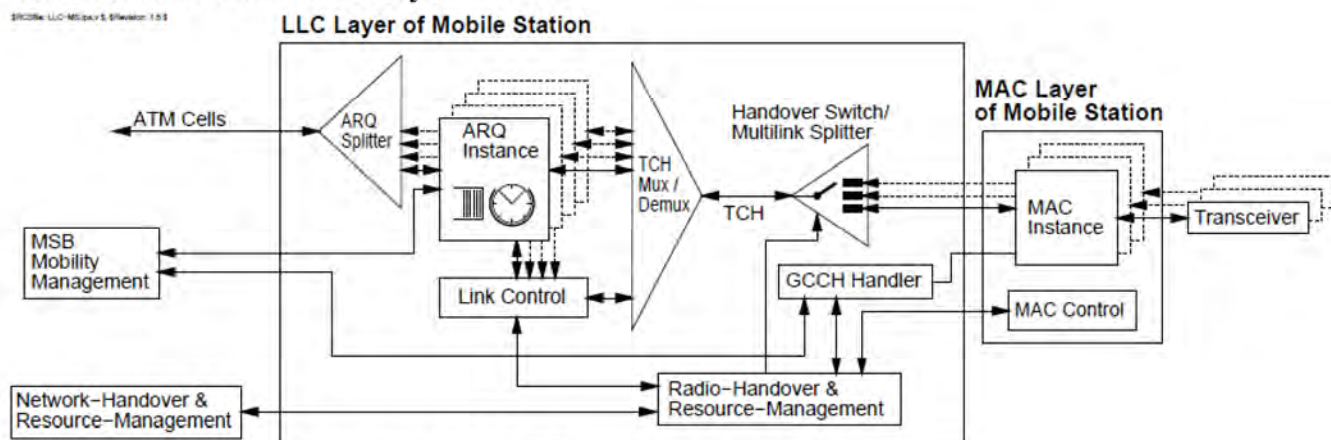


the maximum cell loss rate of the virtual connection to be monitored and to be used to calculate the latest-possible transmission slot for the cell. This minimizes the probability that the cell will be delayed (missed deadline) [39].

The LLC layer can use the following properties of the MAC layer to its advantage:

- ARQ frames are not saved temporarily in the MAC layer. Instead, they are generated by the LLC layer directly before their transmission. This results in a deterministic runtime of the ARQ frame between the partner ARQ instances in the mobile and base stations.
- Since the MAC layer realizes static multiplexes in the physical channels, need-based transmission capacity is available for the ARQ instances, which they can use dynamically.
- Due to the transmission of ATM cells, all information frames have the same length.

#### 4.4. Structure of the MAC Layer



**Figure 4.2:** Layout of the LLC Layer in the Mobile Station

The individual instances of the MAC layer can be seen in on the right side of Figure 4.2. Each MAC instance is assigned a transceiver. The *Global Control Channel (GCCH) Handler* provides a connectionless service as soon as the mobilization has been synchronized with the frequencies of the base station transmission/receiver device. The *Handover Switch/Multilink Splitter* provides a connection-based service and is used primarily for transmitting use data. If connections have not already been established, the signaling for establishing connections is done using the GCCH Handler. Either way, the service is an unsecured service, i.e. there is no guarantee that the transmission will be successful.

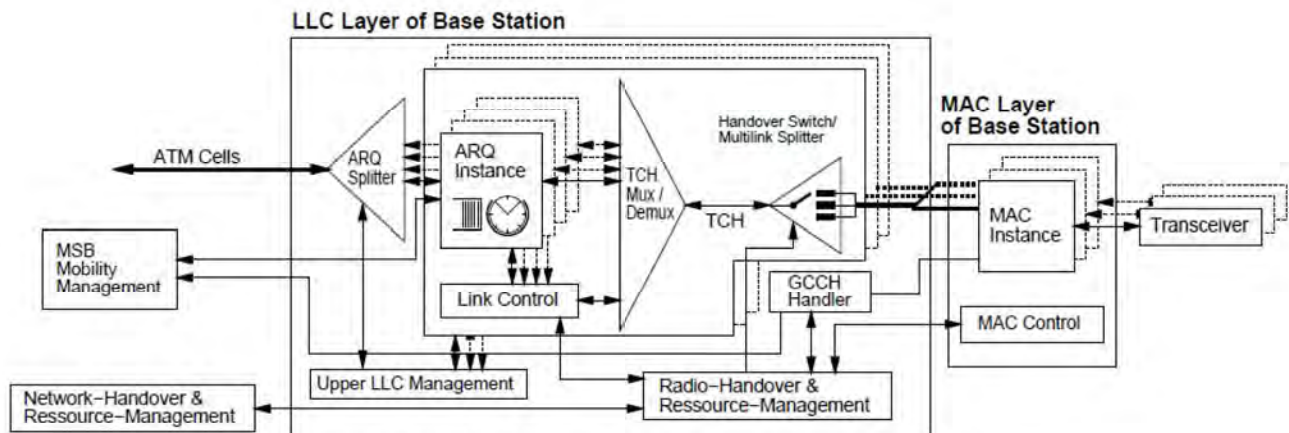
#### 4.5 Structure of the LLC Layer

In Figure 4.2 the LLC layer is shown next to the MAC Layer. The LLC layer can be split into an upper and a lower section. On the one hand, the border lies on the *Traffic CHannel*

shown in the figure as TCH; on the other hand, there is a split between the *Radio Handover & Resource Management* and the *Link Connection*.

In the mobile station, the lower LLC layer consists mainly of the *GCCH Handler*, the *Handover Switch/Multilink Splitter* and the *Radio Handover & Resource Management*. The *GCCH Handler* distributes the data received from the MAC layer and forwards it to either the *Radio Handover & Resource Management* or to the upper layer. The *Handover Switch/Multilink Splitter* distributes the data it received from the *Traffic Channel* among the individual MAC instances. The *Radio Handover & Resource Management* makes statements on which data volumes should be sent to which connections. The allocation of the data volumes is done in the *Radio Handover & Resource Management* of the *Base Station Controller*. The CAC algorithms should be stored here as well. Moreover, the *Radio Handover & Resource Management* is responsible for preparing the handover and managing the connections. It makes the decisions on establishing signaling connections to the *Base Transceiver Stations* or other *Base Station Controllers*.

The ARQ instances can be found in the upper left section of Figure 4.2. In the *ARQ Splitter* the ATM cells are merged upwardly in a single connection to form an ATM cell flow to the next-highest layer. The *ARQ Splitter* converts the long number of the virtual channel into a short address (LLI = *Logical Link Identifier*) so that the number does not have to be transmitted as well. Downwardly, the data that will be sent via the *Traffic Channel* are merged into the lower layer by a multiplexer (*TCH Mux/Demux*). The *ARQ Instance* is selected based on the priority of the different cells at the time of transmission. There is no further temporary storage of the protocol units in the lower sub-layer of the LLC, which would only lead to additional delays and a less dynamic protocol behavior. The *Link Control* ensures the assembly and disassembly of the ARQ instances. It also makes statements on which ARQ instances are assigned which virtual channels and when they should be assembled and disassembled. To do so, it requires information from the network layer located above the LLC layer. A special ARQ instance is parked for the internal MBS signaling and its service is provided by a special SAP of the network layer.



**Figure 4.3:** Layout of the LLC Layer in the *Base Station Controller*

Figure 4.3 shows the structure of the LLC in the *Base Station Controller*. The upper section of the LLC layer contains an equivalent of the LLC layer of the mobile station for each registered mobile station. I.e. for each mobile station that exchanges data with the *Base Station Controller* there is a *Traffic Channel (TCH)* to which the lower LLC sub-layer forwards the data it has received. There is also a *Link Control* that manages the individual ARQ instances for each

of these blocks that is responsible for a mobile station. Only the *ARQ Splitter* works in a generally manner, working across all of the blocks and distributing the ATM cells to the ARQ instances by mobile stations and virtual channels. The upper LLC layer is managed in the *Base Station* by the *Upper LLC Management*. It initiates the assembly and disassembly of the individual blocks.

There is a *Handover Switch/Multilink Splitter* that is equivalent to each *TCH*. There is only one *GCCH Handler* and *Radio Handover & Resource Management*, just as in the mobile station. However, there are clear differences between the interface to the MAC layer and the interface to the LLC layer. In this case, the connection end points are addressed hierarchically in two steps. Please see [6] for details.

## Security Protocols

Security protocols are responsible for mastering transmission errors. They were developed to meet the following requirements [40]:

- Uniform format for transmitting control and message data
- All transmission errors are identified with the exception of a very small error rate ( $10^{-6}$ - $10^{-9}$ ).
- Data blocks are neither lost, nor are they sent repeatedly or swapped.
- Support of standard structures such as point-to-point, multi-point and ring by a single protocol

There are two different classes of approaches for increasing the error safety of a transmission system:

### **FEC ( Forward Error Correction )**

With the FRC approach the decoder in the receiver must be able to not only identify errors, but to localize the error sites as well so that the erroneous bits can be corrected. To do so, the received test bits are compared to the calculated test bits.

### **ARQ ( Automatic Repeat ReQuest)**

With the ARQ approach, the decoder in the receiver merely has to identify errors. When an error occurs, a reverse channel automatically sends a request for a new frame to replace the erroneous frame. The frame is then transmitted again. In contrast to the FEC, any frames received must be acknowledged in the ARQ, which creates an additional load for the reverse channel.

Coding theory states that the number of errors that can be corrected, with the same number of test bits, is much smaller than the number of errors that can be identified. On the other hand, the acknowledgements of the ARQ approach create an additional burden for the reverse channel. A hybrid approach limits the coding effort for the FEC and reduces the number of new requests sent by the ARQ. This type of hybrid approach should be used in the MBS.

This does require a mechanism for identifying received ARQ frames that are erroneous. Please refer to the references for information on a possible combination of the ARQ protocol with an FEC approach within the physical layer [33].

## **5.1 Conventional ARQ Protocols**

The following conventional ARQ protocols will be introduced in the following:

1. Stop-and-Wait ARQ

2. Go Back  $n$  (continuous) ARQ
3. Selective Repeat (SR) ARQ

There are two main evaluation criteria:

### Accuracy

How can it be ensured that all frames are forwarded in the right order and without errors or repetitions in the receiver to the upper layers?

### Efficiency

How much additional channel capacity is used or wasted?

To simplify things, it is assumed that an error identification algorithm is in place and that it will identify any errors that occur. The accuracy can only be shown under these assumptions. Unidentified erroneous frames can lead to erroneous data. Moreover, the frames must be received in the same order that they are sent. The synchronous channel in the MBS ensures that this requirement is met.

#### 5.1.1 Stop-and-Wait ARQ

This is the simplest of all of the ARQ protocols and is based on the idea that the correct receipt of a frame is confirmed before the next frame is transmitted.

To ensure the uniqueness of the frame and acknowledgements when transmission errors or overlapping occurs, both the frame and the acknowledgements must be numbered. These numbers are referred to as *sequence numbers (SN)* for frames from the sender and as *request numbers (RN)* for acknowledgements from the receiver.

Before transmitting a frame, the sender registers the current sequence number, starts a timer and waits for an acknowledgement. If the timer runs out before an acknowledgement is received, the frame that was sent is resent and the timer is restarted.

The receiver only accepts the frame if the sequence number of the received frame is the same as the request number ( $SN = RN$ ). If so, a positive acknowledgement is sent; otherwise, a negative acknowledgement is sent. Frames are acknowledged by entering the sequence number of the next expected frame as the request number ( $RN = SN + 1$ ). Acknowledgments can be transmitted immediately upon receipt of a frame, delayed by any amount of time – although not unending – or in the opposing direction along with the use data. The merging of use data and acknowledgments into a single frame is referred to as *piggybacking*. When doing so, it is important to ensure that the acknowledgement will be transmitted after a finite amount of time. If a negative acknowledgement is received, the sender repeats the  $RN$  frame. If a positive acknowledgement is received, the new  $RN$  frame is sent. In other words, an  $RN$  is sent regardless of the acknowledgement type (positive or negative).

According to [2], a full induction can be used to show that the receiver is only forwarding frames in the correct order. The requirement here is that the sender and receiver have been correctly initialized, e.g.  $SN = RN = 0$ . It can also be shown that there is never a deadlock when there is a finite timer in the sender and a finite delay in the receiver. Two states are sufficient for differentiating between the frames and the acknowledgements, so that model 2 can be coded for

$SN$  and  $RN$ . Thus, there is no need to differentiate between positive and negative acknowledgments either [9].

The Stop-and-Wait ARQ is not particularly effective since a large amount of channel capacity is lost while waiting for acknowledgments. Moreover, *each* frame has to be acknowledged individually, which increases the load on the reverse channel.

### 5.1.2 G Back $n$ (continuous) ARQ

The Go Back  $n$  or continuous ARQ protocol is the most commonly used ARQ protocol and is used in the following standards [2]:

- HDLC (**H**igh level **D**ata **L**ink **C**ontrol procedure) by ISO
- ADCCP (**A**dvanced **D**ata **C**ommunication **C**ontrol **P**rocedure) by ANSI
- SDLC (**S**ynchronous **D**ata **L**ink **C**ontrol procedure) by IBM
- LAP-B (**L**ink **A**ccess **P**rocedure-**B**alanced) by CCITT

HDLC and ADCCP are identical in regard to the ARQ protocol, whereas SDLC and LAP-B represent an unlimited subset of HDLC functionalities.

In contrast to the Stop-and-Wait SRQ, up to  $n$  frames can be sent successively with the Go Back  $n$  ARQ without waiting to receive an acknowledgment between transmissions. This is referred to as a window mechanism with a window size of  $n$ . Two additional sequence numbers are needed to control the window.  $SN_{min}$  refers to the sequence numbers that were most recently requested by the receiver;  $SN_{max}$  refers to the number of the frame that will be sent next, i.e.  $SN_{min} \leq SN_{max} \leq SN_{min} + n$ . Once the frame has been sent with the number  $SN_{min} + n - 1$ , the Go Back  $n$  ARQ must wait for an acknowledgement as well, since the window is now closed. Following a timeout, which can be any finite length, frames  $SN_{min}$  to  $SN_{max} - 1$  are repeated. When an acknowledgement with  $SN_{min} \leq RN \leq SN_{max}$  is received, all frames from  $SN_{min}$  to  $RN - 1$  are acknowledged and the transmission window is shifted so that  $SN_{min} = RN$ . When a positive acknowledgement is received, the process continues from frame  $SN_{max}$  and if a negative acknowledgement is received it continues from  $SN_{min} = RN$ .

The receiver behaves in the same manner as with the Stop-and-Wait ARQ (chapter 5.1.1). Even the proof of accuracy follows the same process. However, please refer to the references for details [2].

According to the requirements, the following applies for sequence number  $SN$ :

$$SN_{min} \leq SN \leq SN_{min} + n - 1 \quad (5.1)$$

The following applies for request number  $RN$ :

$$SN_{min} \leq RN \leq SN_{min} + n \quad (5.2)$$

From (5.1) and (5.2) it follows that the following must be true for model  $m$  to ensure unique coding of the sequence numbers:

$$m > n \quad (5.3)$$

If module  $m$  is given, the maximum window size is:

$$n_{max} = m - 1 \quad (5.4)$$

The approach taken after an acknowledgement is received has a decisive influence on how effective the protocol is. It is important to differentiate between throughput and delay. If the process continues from frame number  $SN_{max}$  after an acknowledgement is received, there is no reaction to the missing frame until the transmission window is closed. Moreover, at least  $n$  frames must be repeated if an error occurs. With  $p$  representing the frame error probability, the throughput thus falls to:

$$D \leq \frac{1 - p}{1 + n \cdot p} \quad (5.5)$$

However, if the process continues from frame number  $SN_{min}$ , the delay is minimized and the lost frame can be quickly repeated; however, the throughput is reduced since frames are being repeated which are most likely still being transmitted. Using the *Ignore Timer* helps in this case, as it monitors the transmission time (chapter 5.2.2).

If the signal runtime is too long, as is the case with satellite transmissions, a correspondingly large  $n$  must be selected in order to avoid wait times when transmission windows are closed to the largest extent possible. This, just like poor transmission quality, also reduces the throughput (equation (5.5)).

### 5.1.3 Selective Repeat (SR) ARQ

The main idea behind the SR ARQ is that the receiver accepts frames that it did not expect to be the next frame. These frames are stored temporarily and only the missing frames are requested from the sender. As soon as the missing frames have been correctly received, all of the temporarily saved frames are forwarded to the next-highest layer. Only repeating the missing frames will achieve the maximum possible throughput of:

$$D = 1 - p \quad (5.6)$$

The *Ignore Timer* makes this possible (chapter 5.2.2).

SR ARQ also results in a window of size  $n$  in the receiver. In contrast to the sender (chapter 5.1.2), no further sequence numbers are required in the receiver to identify the receipt window.  $RN$  now refers to the lowest frame that has not yet been correctly received. In other words, the receiver accepts all frames for which the following applies for the sequence number:

$$RN \leq SN \leq RN + n - 1 \quad (5.7)$$

The range of valid sequence numbers can be determined from equations (5.1) and (5.2) with equation (5.7):

$$RN - n \leq SN \leq RN + n - 1 \quad (5.8)$$

From equation (5.8) it follows that the following must be true for model  $m$  to ensure unique coding of the sequence numbers:

$$m > 2 \cdot n - 1 \quad (5.9)$$

If model  $m$  is given, the maximum window size for SR-ARQ is:

$$n_{max} = \frac{m}{2} \quad (5.10)$$

Based on (5.10), the accuracy of the SR ARQ is determined in the same manner as for the Go Back  $n$  ARQ (chapter 5.1.2).

There are multiple ways to request the retransmission of erroneous frames:

- Positive acknowledgements are used if there are no errors. If one or more frames are missing, they are requested using a negative REJ (**REJ**ect), in which  $RN$  indicates the number starting at which all of the frames should be repeated. In this case, the SR ARQ behaves in the same manner as the Go Back  $n$  ARQ and does not offer a significant advantage.
- Positive acknowledgments are sent if the receipt is error-free, at which point the sender shifts his window. Missing frames are requested using an SREJ (*Selective REJ*ect), in which  $RN$  indicates the number of the missing frame. The sender only re-transmits that specific frame. If multiple frames are missing, each one must be requested individually.
- A field is sent in the acknowledgment that indicates which frames were sent successfully and which contained errors. The sender can then simply retransmit the missing frames. These acknowledgements can create considerable overhead depending on the size of the window.

The individual acknowledgement types do not exclude one another and can be combined to create the highest possible efficiency among the requests.

## 5.2 The Adaptive Selective Repeat (ASR) ARQ Protocol

The ASR ARQ protocol [10] is based on the SQ ARQ. The properties of the ASR ARQ can be divided into three areas:

1. Elements from the standard ISO 8802-2
2. Handling time-critical ATM cells
3. Advantages of parallel ARQ instances

The individual properties are described in detail in the following.



### 5.2.1 Elements of the ARQ in the Standard ISO 8802-2

The ARQ protocol specified in ISO 8802-2 (IEEE 802.2) serves as the basic protocol and is a modified Go Back  $n$  ARQ with the following elements [17]:

#### Acknowledgement Timer

The Acknowledgement Timer monitors the time in which an acknowledgement is expected once one or more frames have been sent. If an acknowledgement is not received, the sender begins to *poll*.

#### P-Bit Timer

The P=Bit Timer monitors the time in which an answer to a *poll* is expected. The poll is repeated if an answer is not received.

#### Reject Timer

The Reject Timer monitors the time in the receiver in which an answer to a new request is expected. The request is resent if an answer is not received.

#### N2

N2 indicates the maximum number of repetitions if an Acknowledgment, P-Bit or Reject Timer runs out. The virtual channel is reset if N2 is exceeded.

#### Receive Ready (RR) Protocol Data Unit (PDU)

The RR DPU is used to send positive acknowledgements, i.e. all frames are acknowledged for which the following is true:  $SN < RN$

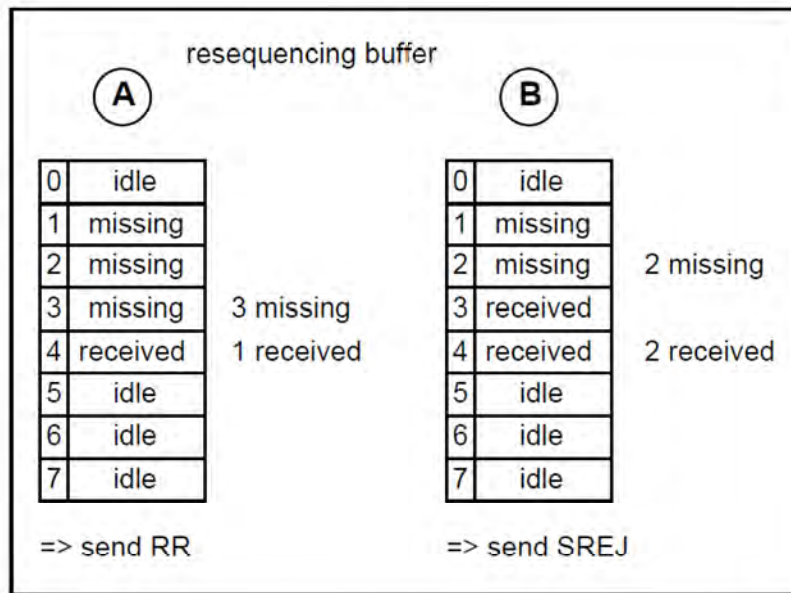
#### REject (REJ) PDU

The REJ PDU is used to send negative acknowledgements, i.e. acknowledgements are issued in the same manner as the RR PDU and all frames are repeated for which the following is true:  $SN \geq RN$

Furthermore, the standard provides measures for *flow control*. The LLC layer of the MBS does not have *flow control*, which is why these measures will not be examined in detail herein.

The ASR ARQ inherits all of the above-mentioned elements. The *Acknowledgement Timer* is required for the stability of the protocol. Both the *P-Bit* and the *Reject Timer* are measures used to reduce frame delays. The effectiveness of the three timers is examined using simulations (chapter 8), which is why they were implemented in a manner that allows them to be activated and deactivated. The *Timer Delays* of all three timers are free parameters, which must be adjusted to the respective conditions.

The ASR ARQ is a modified SR ARQ and uses a *Selective REject (SREJ)* PDU, which itself is used to request an individual frame again. Similar to the REJ PDU, all frames where  $SN < RN$  are acknowledged, but only the frames where the number is  $SN = RN$  are resent. With ASR ARQ, the RR PDUs are interpreted in manner similar to the REJ PDUs in the ISO 8802-2. This approach is only reasonable when used in combination with the *Ignore Timer* and REJ PDUs are not used in this case. If the receipt buffer contains fewer frames than the number of frames that are missing in the receipt order (Figure 5.1, Case A), an RR PDU will be sent instead of requesting each frame individually using SREJ PDUs. Otherwise, the missing frames will be requested individually using SREJ PDUs (Figure 5.1, Case B).



**Figure 5.1:** Example of the Sending of RR and SREJ

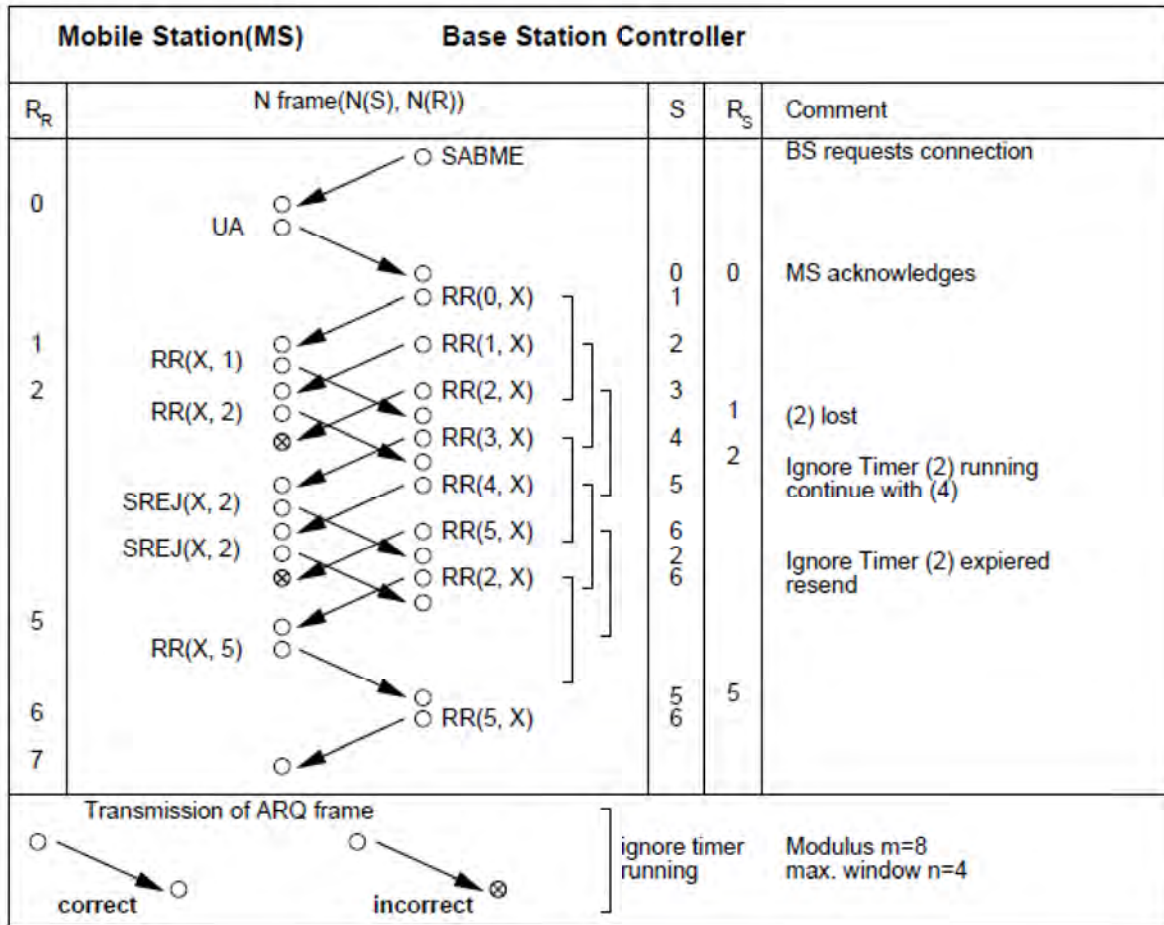
### 5.2.2 The Ignore Timer

One problem with ARQ protocols is that when the sender receives an acknowledgement he/she is unable to determine what happened to the frames that were sent when the acknowledgment was being transmitted. It is certainly possible that the requested frames have already been resent, but simply had not arrived in the receiver when the acknowledgement was sent, particularly when a new request is received from SREJ or REJ PDUs. However, if the transmission time could be determined, it would be possible to determine which frames simply had not yet arrived in the receiver. The transmission time is deterministic in the MBS. On the one hand, the SRQ protocol generates PDUs directly before sending them; while on the other hand, the signal runtime in the MBS is limited due to the restricted cell size. The total transmission time fluctuates between one and two slots depending on the shift in the uplink and downlink.

The Ignore Timer uses this time, which is referred to in the following as *Ignore Delay*. The Ignore Timer is started before sending a PDU. Any new requests for frames are ignored as long as the Ignore Timer is running. This avoids unnecessary repetitions.

If the sender receives a positive acknowledgement with  $RN < SN$  and if the Ignore Timer for frame  $SN$  has already run out, the sender knows that the frame was not received correctly and he/she can resend it. This approach has an informational advantage over the SR ARQ, since the sender can initiate repetitions *before* new requests are received when using the ASR ARQ. The advantage is that the receiver is not informed that a frame was lost until the subsequent frame is correctly received. Until then, a positive acknowledgement does not lead to a repetition of the lost frames in SR ARQ.

If the receiver only uses SREJ PDUs to request missing frames, the Ignore Timer can be used to achieve the optimal throughput of  $D = 1 - p$ .



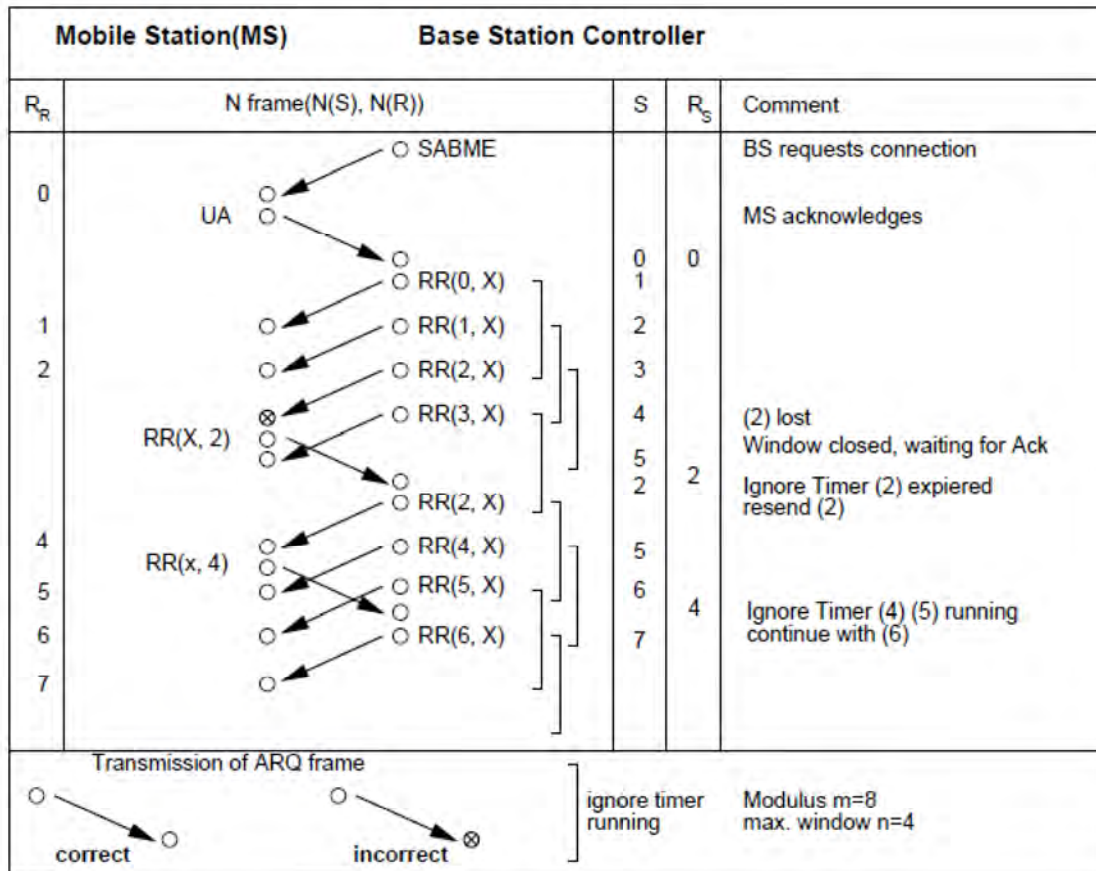
**Figure 5.2:** Avoiding Unnecessary Repetitions by Using the *Ignore Timer*

In summary, it can be said that the Ignore Timer has two definite advantages:

1. Unnecessary repetitions can be avoided, the optimal throughput can be achieved
2. Faster identification of lost frames, reduced cell delay

Two examples are provided that demonstrate how the Ignore Timer works. Figure 5.2 examines a channel with unidirectional communication. All cells are time-critical, meaning the receiver must issue acknowledgements more quickly, i.e. the receiver sends an acknowledgement as soon as a frame is received.

The base station (BS) uses an *SAMBE* (**S**et **A**synchronous **B**alanced **M**ode **E**xtended) PDU to request a connection. The request is confirmed by the mobile station (MS) using an *UA* (**U**nnumbered **A**cknowledgment). The BS then sends frames 0-2 sequentially, but the last one is lost. The receipt of frame 0 is confirmed when *RR(X, 1)* is received and the BS shifts the transmission window to 1. When frame 3 is received, the MS realizes that frame 2 is missing. It sends an *SREJ(X,2)* PDU to request the missing frame. It repeats the same process when frame 4 is received. Since the *Ignore Time* for frame 2 has already run out, it is resent as a request for it was received and the timer is restarted. The second request is ignored. Once *RR(X, 5)* is received, the BS can shift the window to 5 and transmit the requested frame. The protocol then continues normally.



**Figure 5.3:** Fast Reaction to Losses Using the *Ignore Timer*

Figure 5.3 also examines a channel with unidirectional communication and a time-critical service. However, the reverse channel only has limited capacity, meaning the receiver can only send acknowledgements every few time slots.

The BS uses an *SAMBE* PDU to request a connection. The MS confirms the request using an *UA* PDU. The BS then sends frames 0-2 sequentially, although the last one is lost. The transmission window is closed after frame 3 is sent and the BS waits for an acknowledgement. By the time the MS can send an acknowledgement, it is not yet aware that frame 2 was lost. As a result, it sends an *RR(X, 2)*. The Ignore Timer has already run out in the BS, which means that by the time the acknowledgement is received the BS is aware that frame 2 was lost. Frames 0 and 1 are acknowledged, frame 2 is repeated, while frame 3 is not flagged to be resent since the Ignore Timer is still running. The BS correctly assumes that frame still being transmitted will be correctly received. By skipping frame 3, it then sends frames 4 and 5. The process can continue normally once acknowledgment *RR(X, 4)* is received.

Without the use of the Ignore Timer, frame 2 would not have been resent when the *RR(X,2)* acknowledgment was received. It would have been necessary to request it individually using a negative acknowledgment (SREJ, REJ). The limited capacity of the reverse channel would have been further burdened by another acknowledgment. Moreover, the additional need for acknowledgment would have increased the probability that the time window would have closed again and the sender would have had to sit and wait for an acknowledgement.

### 5.2.3 Handling Time-Critical ATM Cells

ATM uses a QoS parameter for the maximum delay for time-critical services. If adherence to that parameter is more important than adherence to the maximum cell loss rate, the ASR ARQ has mechanisms for ensuring the maximum delay is not exceeded, which it does by discarding specific cells when there is a high load. This type of service could include a video conference connection, where the loss of individual cells is not significant, but where the maximum delay must be adhered to due to the synchronizations and echo effects.

Three measures are provided to discard cells:

1. Cells from the first transmission are discarded once they have exceeded the maximum delay. These cells are not transferred to the transmission window, i.e. are removed from the transmission window and replaced with newer cells. This is the only case where the receiver does not need to be informed of the discard, before the first transmission.
2. A *Delay Timer* is set in the sender to control the remaining lifecycle of the cells in the transmission window. If the time expires, the cell is discarded. If the cell was already being transmitted, the receiver must be informed that it was discarded. This information is transmitted using a *Delay PDU* (chapter 5.2.4).
3. The remaining lifecycle of cells in the receiver that had to be stored temporarily as the receiver was waiting for frames missing in the sequence are monitored. If the time runs out, the wait process is terminated and any cells that have already been received are forwarded to the upper layer. The remaining lifecycle must be transmitted as well for this measure, which means it involves an increased overhead. This measure is only reasonable in connection with measure 2.

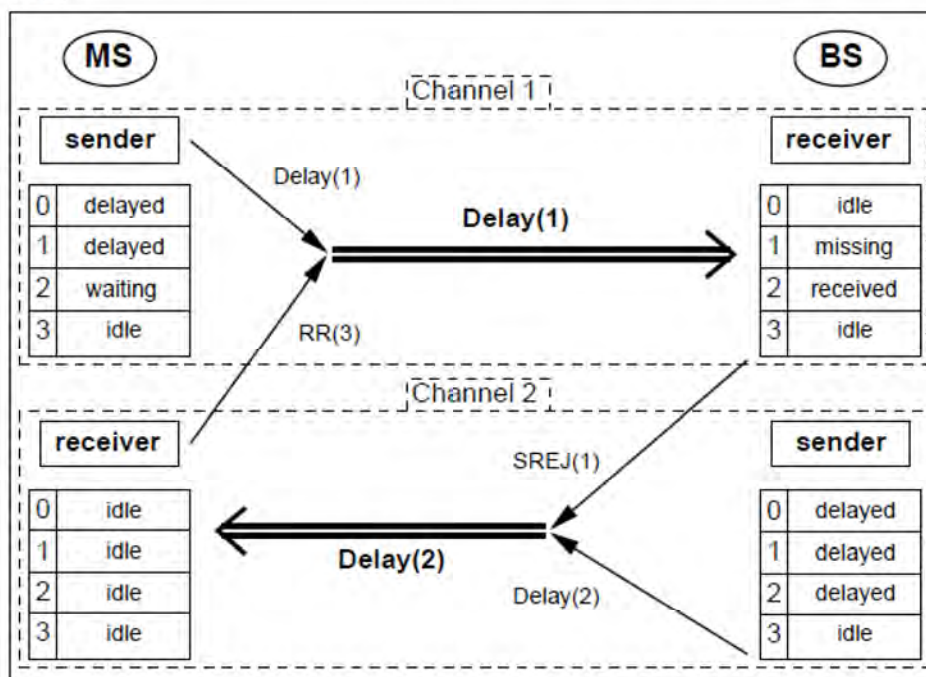
These three measures are consecutive, building on one another, i.e. measure 1 is required for measure 2. Strict adherence to the maximum delay can only be achieved by using all three measures; otherwise, individual cells could exceed the maximum delay. These measures increase the complexity of the protocol considerably. Accuracy is no longer a given and exceptions must be considered as certain conditions could lead to a deadlock.

### 5.2.4 The *Delay PDU*

The *Delay PDU* is used to inform receivers that cells have been discarded. It is only sent if the receiver requested a discarded cell (using RR or SREJ). In this case, the *Delay PDU* is sent in the opposite direction instead of an acknowledgement and  $RN = SN$ , in which  $SN$  is the highest number of all of the discarded cells. Prerequisite for doing so is that the sender discard the cells in the proper order, i.e. there cannot be valid (not discarded) cells with lower sequence numbers. For this reason, the number of the most recently discarded cell has the greatest informational value and may, under certain conditions, be sent instead of the requested cell.

If the receiver receives a Delay PDU, it stops waiting for cells where the following applies for the number:  $N \leq RN$ . It then shifts the window and issues a corresponding acknowledgement. When using bi-directional communication it is important to remember that the Delay PDU competes with the acknowledgments. It must be ensured that both acknowledgments, as well as Delay PDUs can be transmitted within a finite time.

Figure 5.4 shows the deadlock that may occur if the Delay PDU has priority over the acknowledgments:

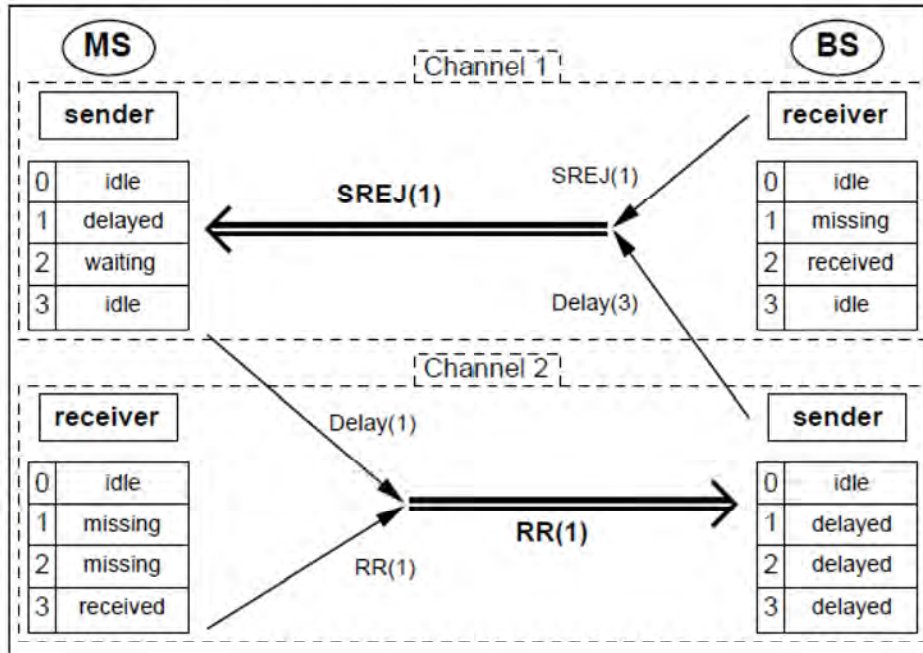


**Figure 5.4:** Deadlock Resulting from Delay PDU Prioritization

In channel 1, the receiver in the BS is waiting for frames (1) and would thus like to send an SREJ(1). However, since the sender in the BS (channel 2) has a Delay(2) with a higher priority that is sent instead. The sender in the MS (channel 1) cannot, however, continue until it has received an acknowledgement. The same is true for the sender in the BS (channel 2), which cannot shift its window until it has received an acknowledgement. If the sender continues to permanently transmit its Delay PDUs, a deadlock arises, which cannot be resolved until a reset is triggered by a timeout.

A deadlock arises in the other extreme as well, where the acknowledgement has priority over the Delay PDU, as shown in Figure 5.5:

In this case, the receiver in the BS uses an SREJ(1) in channel 1 to request the frames (1). This was discarded, but since the receiver in the MS (channel 2) has also requested frames using an RR(1), the cell discard notifications (Delay PDUs) cannot be transmitted. In this situation, neither the sender nor the receiver can shift their window. If the receivers continue to behave in the same manner, a deadlock is created that can only be resolved by a reset. Thus, it must be ensured that the priorities change dynamically and in a manner that ensures both acknowledgements and Delay PDUs can be transmitted within a finite time. The simplest case is to



**Figure 5.5:** Deadlock Resulting from Acknowledgment Prioritization

toggle back and forth between acknowledgement and Delay PDU, i.e. an acknowledgement is sent, followed by a Delay PDU and vice-versa as needed.

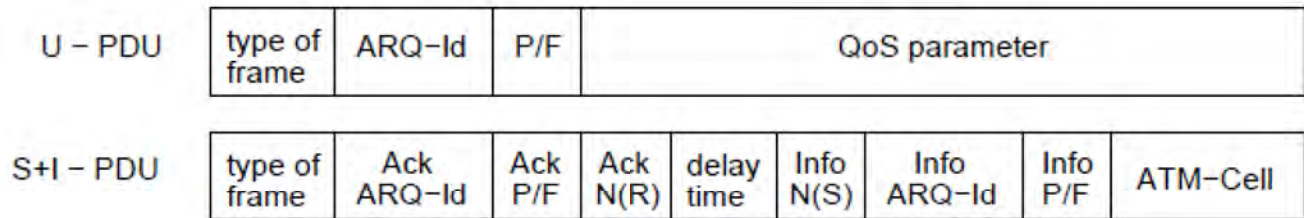
Furthermore, it is important to ensure that either the space in the transmission window in the sender that is occupied by discarded cells is made available or that the Acknowledgement Timer runs after the discard. Otherwise, the discarded cells may occupy the entire transmission window, in which case the receiver will not expect any cells and the sender will not poll, since the timer cannot run out. This creates a deadlock that cannot be resolved, not even by resetting virtual channel.

If the receiver terminates the wait for missing cells, it will shift its window. If the missing cells are received after the window has been shifted, they must all be discarded. They can be identified by the fact that their sequence number is outside the receiver window (which is another reason the window size cannot exceed half of the mode).

All three measures for handling time-critical ATM cells are included in the simulations (chapter 8).

## 5.2 Parallel ARQ Instances

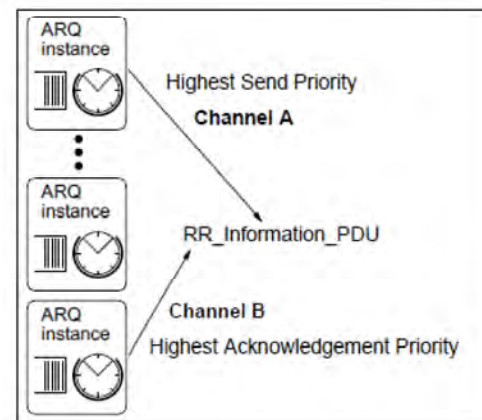
Parallel ARQ instances are constructed in the upper LLC layer of the MBS in order to be able to react properly to the different QoS parameters of the VCs. In contrast to the ATM where cells are served based on a FCFS (First Come First Serve) principle, in ASR ARQ time-critical cells are given priority. A priority algorithm is used which calculates the urgency of the individual cells. Further PDU formats are used to ensure that all the advantages offered by parallel ARQ instances are used.



**Figure 5.6:** Frame Structure of the ASR ARQ

As shown in Figure 5.6, acknowledgements and use data are sent in a frame. This approach is referred to as a piggyback approach. The disadvantage of the original form of this approach is that acknowledgments are dependent on the use data. The priority of the acknowledgment is linked to the priority of the related use data. However, the character of the acknowledgment within a channel is different than that of the use data, especially when working with asymmetrical communication. In this case, acknowledgements are sent late or not at all, or a priority algorithm has to be developed that takes the problem into consideration.

The ASR ARQ uses another method; it adds a second *VC-Id* (channel ID) to the frame structure. This allows use data from channel **A** to be combined with an acknowledgment from channel **B** (Figure 5.7). As a result, acknowledgments and use data from any channel can be combined within a mobile station. The most urgent acknowledgment can then be calculated by a different algorithm independent of the volume of the use data. The advantages are seen most clearly in extremely asymmetric communication within a mobile station. The disadvantage is the generation of additional overheads by the fields *Ack ARQ-Id* and *Ack P/F* (Figure 5.6). The second *P/F* field was added to ensure the polling was could be done as fast as possible.



**Figure 5.7:** Generating Piggyback PDUs

### 5.2.6 Acknowledgement Approach

The following will explain when a receiver would want to send an acknowledgement. Basically, there are different approaches for doing so:

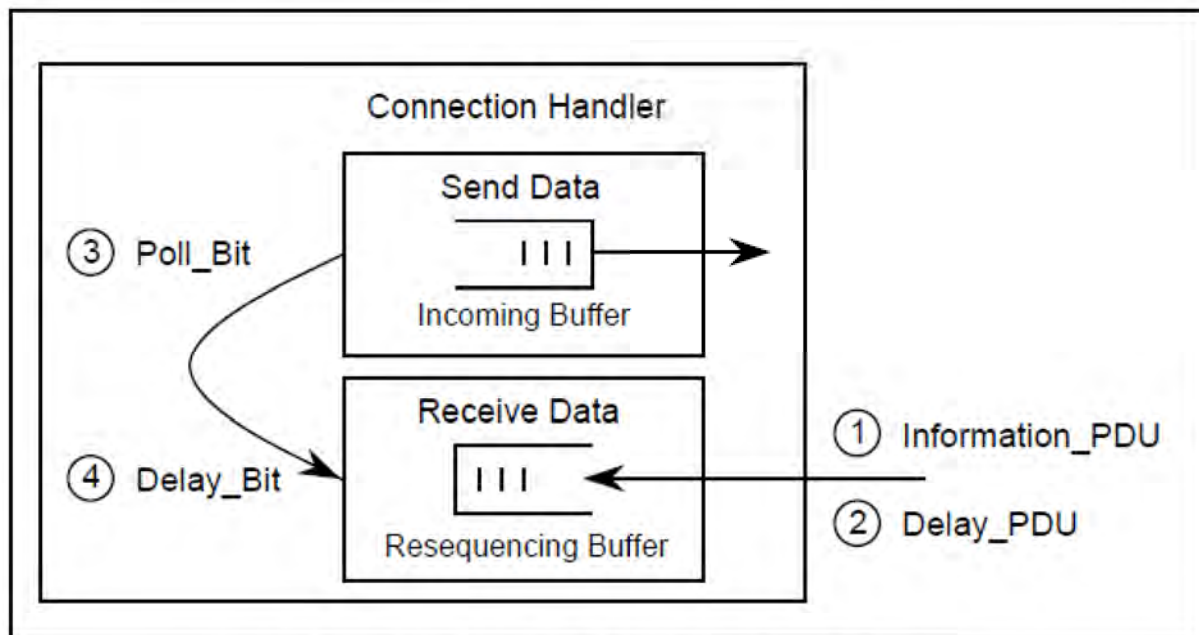
#### Time-based

In this approach, the receiver sends an acknowledgement in equidistant time intervals. This ensures that the sender receives information about the state of the receiver in constant time segments. Otherwise, there would be a higher overhead – especially when communication is sent in clusters – since many of the acknowledgments would have been unnecessary.



### Need-based

Only events that make an acknowledge necessary are reacted to. In other words, acknowledgments are only issued when the receiver identifies a need for one. The overhead is thus significantly smaller compared to the time-based approach, although it can no longer be ensured that the sender will be informed in fixed blocks of time. The following demonstrates the need-based strategy and explains which events trigger acknowledgements (see Figure 5.8).



**Figure 5.8:** Events for Triggering Acknowledgements

1. Receipt of an *Information\_PDU*
2. Receipt of a *Delay\_PDU*
3. Sender sets the *Poll\_Bit* in the corresponding *ARQ Instance*. The sender sets the *Poll\_Bit* that is actually assigned to his use data when he is unable to send data (e.g. due to a closed window), but would like to poll. A second *Poll\_Bit* is provided in the PDUs for that purpose.
4. Corresponding sender sets the *Delay\_Bit* in the same *ARQ Instance*. The sender sets the *Delay\_Bit* when he needs to send a *Delay\_PDU*. The protocol treats the *Delay\_PDUs* as a type of acknowledgment that informs the receiver about the changed state of the sender.

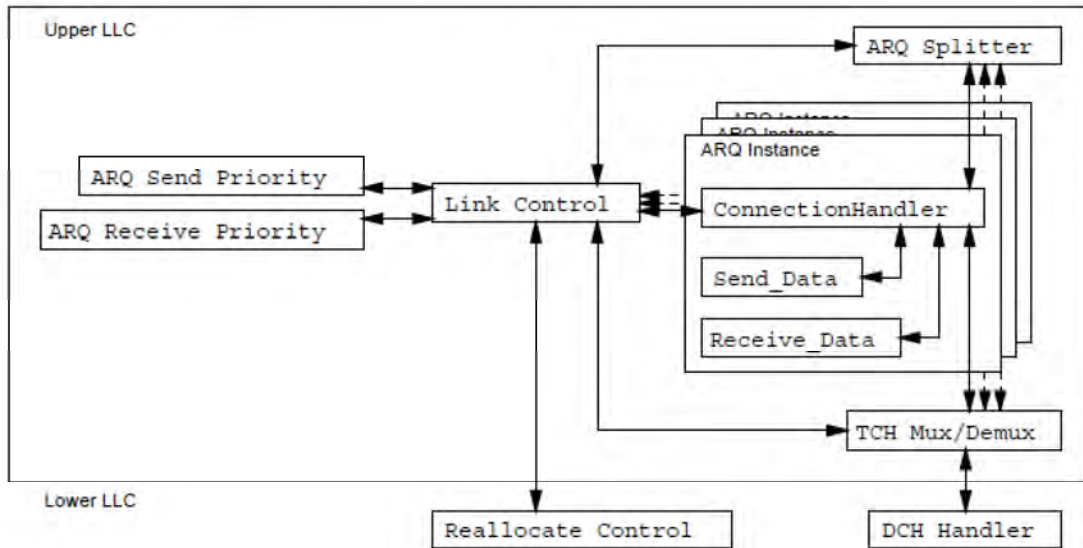
If there are multiple acknowledgement instructions, they must be assigned priorities. The priority can be calculated as follows:

- The acknowledgements are selected based on their time stamp so that the oldest one is sent first.
- The age of the acknowledgment is weighted together with the sender's *maximum delay*.
- The acknowledgments are weighted according to the number of acknowledged PDUs, although the weighting of the *RR* and *SREJ* PDUs is based on the number of received and missing PDUs.

- The acknowledgment types are processed in the following order:
  1. Polled acknowledgement
  2. Delay
  3. Selective Reject (SREJ)
  4. Receive Ready (RR)

Acknowledgements can be collected to minimize the protocol overhead, thereby reducing the absolute number of acknowledgements. Acknowledgments can also be delayed, in order to send them together with ATM cells. Priority thresholds are defined in the protocol for that purpose. The acknowledgements are not sent to an empty information frame until they have a priority higher than the threshold, in which case they may be sent without ATM cells as well.

## Implementing the ASR ARQ Protocol



**Figure 6.1:** Structure of the Upper LLC Layer

Figure 6.1 shows the structure of the upper LLC layer in the MBS simulator. This layer provides an interchangeable module within the protocol stack of the simulator and can, for the most part, be operated independent of the other modules. The exact configuration is described in chapter 7.

During the initialization phase of a station, the **Link Control**, **ARQ Splitter**, **TCH Mux/Demux** and the priority objects are constructed and remain in place throughout the entire operating period. The ARQ instances, on the other hand, are assembled and disassembled as needed.

The **ARQ Splitter** is the central object of the upper LLC layer. It assembles and disassembles channels by assembling and disassembling ARQ instances. The **Link Control** also manages the assignment of ARQ instances to virtual channels. The entire inner-layer management communication runs through the **Link Control** and all other objects have access to it. If the **Link Control** receives a connection request from the upper layer, it forwards the request to the **Reallocate Control** to determine whether there is sufficient capacity and if it receives a positive answer, it assembles an ARQ instance with the requested QoS parameters. The new channel is registered with the priority objects. The process is reversed if a connection is released.

The **ARQ Splitter** distributes the ATM cell flow among the individual ARQ instances. When doing so, it evaluates the VC ID and, in the base station, the Mob (mobile station) ID and uses them to request the corresponding ARQ instance from the **Link Control**. If it is available, the data is forwarded to the **ConnectionHandler** of that instance. Data from the **ConnectionHandler** is passed to the higher layer.

When working with data from the **DCH Handler**, the **TCH Mux/Demux** behaves in the same manner as the **ARQ Splitter** and forwards data to the **ConnectionHandler** of the responsible ARQ instance. The current state is then determined by the **Link Control** by activating the corresponding function of the priority objects. To generate a PDU, the ARQ instance that most urgently needs to send use data (**ARQ Send Priority**) is determined and the **ConnectionHandler** of that instance is ordered to generate an Information PDU. Next, the ARQ that most urgently needs to send an acknowledgement (**ARQ Receive Priority**) is determined. If necessary, that instance assembles a Piggyback PDU from the Information PDU and the acknowledgement. The **TCH Mux/Demux** returns the Piggyback PDU to the **DCH Handler**. If a connection request is received, it is forwarded to the **Link Control**, which constructs an ARQ instance. It does not need to ask the **Reallocate Control**, as that released the channel earlier on when the request was received in the partner instance. The connection request is then transferred to the **ConnectionHandler** of the new instance for further processing. The process is reversed when there is a request to release a connection, i.e. the request is processed in the **ConnectionHandler** first, and then the ARQ instance is assembled by the **Link Control** if so requested by the **TCH Mux/Demux**.

Priority algorithms **ARQ Send Priority** and **ARQ Receive Priority** have two tasks:

#### **Calculate the Status of the ARQ Instances**

The number of send requests (**Length**), the oldest date (**First\_Gen\_Time**) and the shortest remaining lifecycle (**Remain\_Life\_Time**) are calculated. The ARQ instances are accessed using the **Link Control** to determine the data. Although the algorithms for the sender and receiver side can be very different, the interface is the same. This makes it easier to implement new algorithms.

#### **Calculate the Send Priority**

The ARQ instance that wishes to send data is determined here. In doing so, the algorithm for determining the highest priority needs to follow the same rules as those used to determine the status. Thus, these two tasks were merged in the priority objects. In contrast to send algorithms, the receive algorithm can also provide information that no instance needs to be given priority. In this case, the acknowledgement for the instance that is sending use data is transmitted.

The **Short\_Remain** algorithm was implemented as the **ARQ Send Priority**. The cell with the shortest **Remain\_Life\_Time** is given the highest priority. **Length** and **First\_Gen\_Time** are not taken into consideration in this algorithm.

Three algorithms were implemented as the **ARQ Receive Priority** for comparison:

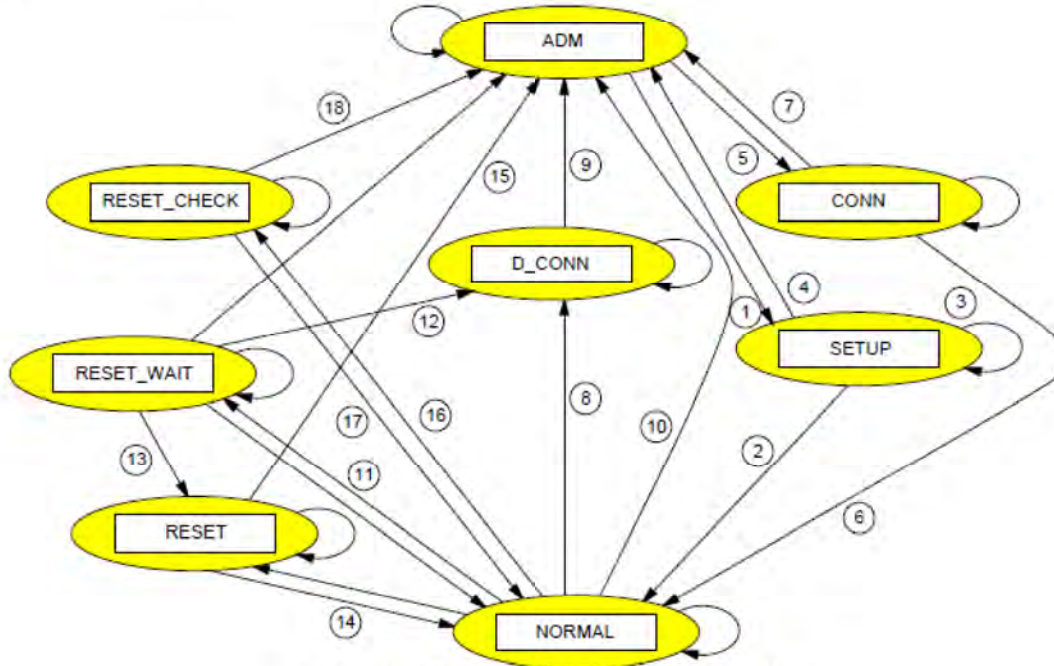
**oldest\_first** This variant considers the **First\_Gen\_Time** of the acknowledgments and gives the oldest the highest priority. The number of frames to be acknowledged is also taken into consideration.

**answer\_poll\_first** Answers to polls and Delay PDUs are immediately assigned a higher priority, other acknowledgements are handled using the **oldest\_first** method.

**prefer\_poll** This variant differentiates between the type of acknowledgement and the number of frames to be acknowledged compared to the maximum window size. Answers in the poll cycle and Delay PDUs are favored. The priorities are converted to **Remain\_Life\_Time** in order to compare them to the send priority.

**ARQ Send Priority** and **ARQ Receive Priority** must be coordinated to ensure they deliver comparable priorities, which also ensures acknowledgements can be given priority over use data. This is particularly important when working with asymmetrical communication.

### 6.1 The ConnectionHandler



**Figure 6.2: ConnectionHandler States**

The **ConnectionHandler** is the central object of the ARQ instance. It is implemented as a finite state machine and can assume any of the states shown in Figure 6.2. The state machine is implemented according to ISO 8802-2 in regard to connection establishment or termination and reset handling [17].

Initially, the **ConnectionHandler** is in an inactive state ADM (Asynchronous disconnected mode). The state transitions can be categorized into three groups:

#### Connection Establishment

If the own station requests a connection, it sends an *SABME PDU* and transitions into SETUP state (1). An answer is expected here. If it is positive, the **ConnectionHandler** transitions into NORMAL state and can begin transmitting the use data (2). If an answer is not received, the *SABME PDU* is repeated when the timer runs out (3). After N2 attempts or if a negative answer is received, it returns to an ADM state (4). If an *SABME PDU* is received while in the ADM state, the next-highest layer is informed. In this case, the **ConnectionHandler** transitions into CONN (CONNect) state while waiting for an answer from that layer (5). If a positive answer is received, the **ConnectionHandler** transitions to NORMAL state (6). If a negative answer is received, it transitions to ADM state (7).

### Connection Termination

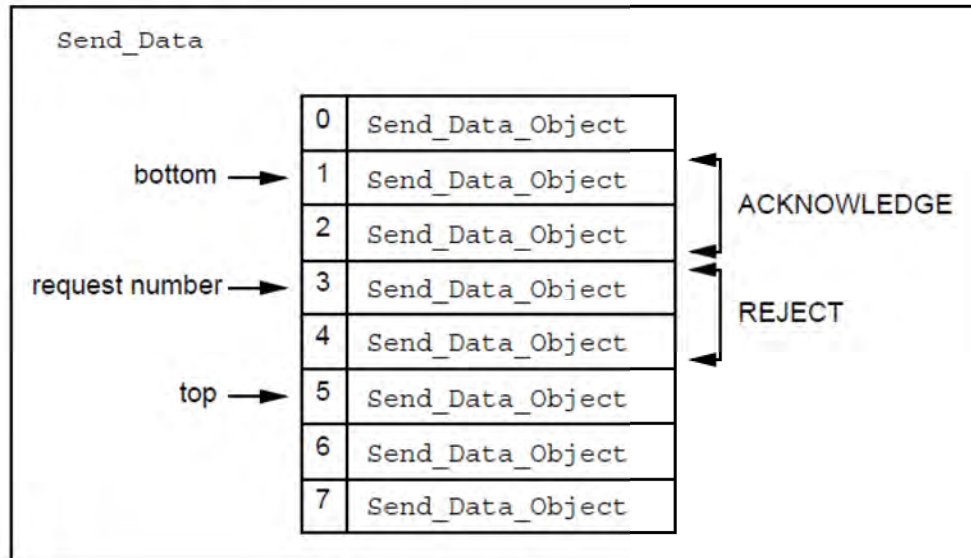
If a connection release request is received while in the NORMAL state, A *DISC (DISConnect) PDU* is sent and the answer is awaited in a D\_CONN (DisCONNect) state (8). Either way, the state is switched to an ADM state (9). The partner instance shows the higher layer the connection release, sends an *UA PDU* and transitions right back into an ADM state (10).

### Reset Handling

If a local reset is triggered when in NORMAL state, the higher layer is informed thereof and the answer is awaited in a RESET\_WAIT state (11). If the connection should be released, the state is transitioned to a D\_CONN state and the process continues as described (12). However, if a reset is performed, the connection is reset, a *SABME PDU* is sent and the answer from the partner instance is awaited in a RESET state (13). If a positive answer is received, the state is transitioned to a NORMAL state (14). If a negative answer it is transitioned to ADM (15). If the partner instance receives the *SABME PDU*, it asks the higher layer how to proceed and transitions into a RESET\_CHECK state (16). If the reset is accepted, the connection is reset on that side, the reset is confirmed by an *UA PDU* and returns to the NORMAL state (17). If the answer is negative, it sends a *DM (Disconnect Mode) PDU* and transitions into an ADM state (18).

In a NORMAL state, control is transferred to the **Send\_Data** and **Receive\_Data** objects, which were generated by the **ConnectionHandler** when the connection was established.

## 6.2 Send\_Data



**Figure 6.3:** Events when an Acknowledgement Is Received

**Send\_Data** manages the transmission window of the ASR ARQ protocol. A **Send\_Data\_Object** is constructed for each window slot, manages the slot independently.

**Send\_Data** can send events to the **Send\_Data\_Object** without having to observe their status, since the objects independently recognize if an event is meaningful for them. This has the advantage that the same process can be used when an acknowledgement is received, regardless of the internal state. **Send\_Data** has three pointers for managing the transmission window:

**bottom** always points to the lowest window position. It also shows the lowest cell that has not yet been acknowledged.

**top** always shows the position to which the new cell was saved. This is only possible if  $top - bottom < n$  ( $n$  is the maximum window size, chapter 5).

**send\_number** shows the cell that should be sent next.

Figure 6.3 demonstrates the acknowledgment mechanism. It shows a **Send\_Data** instance

with eight **Send\_Data\_Object** instances. The maximum window size is  $n = 4$ . The window is closed since  $top - bottom = n$ . **Send\_Data** now receives an *RR PDU* with request number 3, i.e. the receiver expects the next number to be 3. Next, slots *request number - 1* to *bottom* are acknowledged (ACKNOWLEDGE). In this case those are slots 2 and 1 (in that order). Next, all slots from *request number* to  $top - 1$  are rejected (REJECT), in this case 3 and 4. The **Send\_Data\_Object** decides how the acknowledgement and rejection are handled.

Pointers *bottom* and *send\_number* now need to be updated. The *bottom* pointer is increased as long as state of the **Send\_Data\_Object** of the *bottom* is IDLE and  $bottom < top$ . Pointer *send\_number* is increased as long as the state of the **Send\_Data\_Object** of the *send\_number* is not SEND and  $send\_number < top$ . The following is true if the transmission window is empty:

$$bottom = send\_number = top \quad (6.1)$$

However, the following is always true:

$$bottom \leq send\_number \leq top \quad (6.2)$$

Pointer *top* is increased when a new cell is saved and only needs to be updated when slots remain unoccupied after a refresh (see chapter 6.2.1).

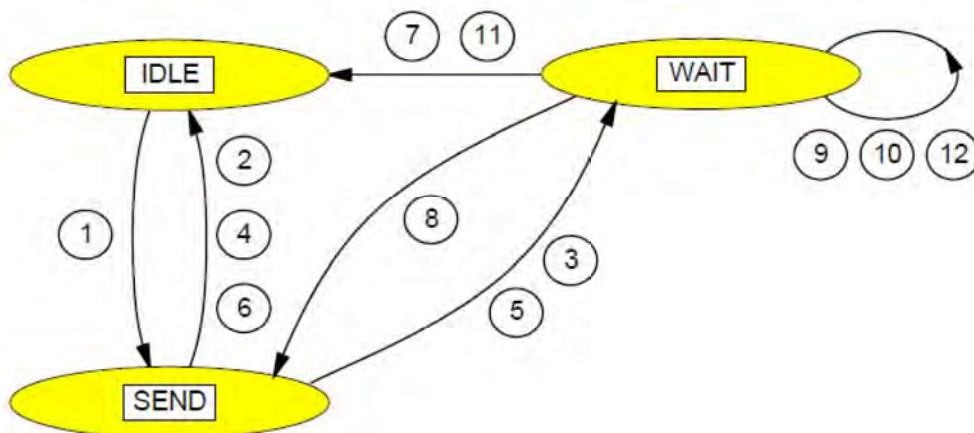
In addition to these tasks, **Send\_Data** must also manage the Acknowledgement Timer, inform the receiver instance when poll bits are set and when Delay PDUs are sent.

### 6.2.1 Send\_Data\_Object

**Send\_Data\_Object** manages a transmission window slot. It is implemented as a finite state machine and can assume the states shown in Figure 6.4:

#### IDLE

**Send\_Data\_Object** is in the IDLE state after the initialization. If a *DATA.request* is received, the SDI (Service Data Unit) is saved, the Delay\_Timer is started and the count is set to *count* = 0. **Send\_Data\_Object** transitions to a SEND state (1).



**Figure 6.4:** States of the `Send_Data_Object`

### SEND

When in a SEND state, the object waits for the SDU to be sent. If the SDU was already sent ( $count > 0$ ) and an *ACKNOWLEDGE* is received, the SDU is deleted and the timer is stopped (2). In case of *TRANSMISSION.indication*, a differentiation must be made between whether the SDU was already sent ( $count > 0$ ) or whether the remaining lifecycle has not yet expired (*delay\_ok*). If the SDU is being sent for the first time and if its remaining lifecycle has expired, the SDU can be discarded and replaced by a newer one. To do so, the SDU is deleted, the object transitions into an IDLE state and a refresh is started (4). Otherwise, the SDU is released for sending, the *ACK\_* or *Ignore\_Timer* is started and the object moves into a WAIT state (3). If the *DELAY\_Timer* runs out while in a SEND state, the SDU is deleted and the object moves into a WAIT state so that the receiver can be informed that the SDU was discarded (5). If, in addition to that, the  $count == 0$ , the receiver does not need to be informed and the object moves into an IDLE state and a refresh is initiated (6).

### WAIT

When in a WAIT state, the object waits for an acknowledgement. If a positive acknowledgement is received (*ACKNOWLEDGE*), the SDU is deleted, the timer is stopped, a *Reset\_Delay* is initiated and the object moves back into an IDLE state (7). In case of a negative acknowledgment (*REJECT*), action is only taken if the *Ignore\_Timer* is no longer running (*NOT Ignore*). If there is an SDU ( $SDU \neq NIL$ ), the *ACK\_Timer* is stopped and the object moves into a SEND state so that it can be resent (8). If the SDU was discarded ( $SDU == NIL$ ), a poll request is explicitly withdrawn (*Reset\_Poll*), a Delay PDU is requested (*Set\_Delayed*) and the WAIT state is maintained (9). The number of transmission attempts becomes relevant if the *ACK\_Timer* runs out. If the count is  $count < N2$ , the *ACT\_Timer* is restarted, the *count* is increased by one, the poll bit is set and the WAIT state is maintained while waiting for an acknowledgement (10). If the count is  $count \geq N2$ , the SDU is deleted, the *Delay\_Timer* is stopped, a local reset is triggered and the object moves back into an IDLE state (11). If the *Delay\_Timer* runs out, the SDU is deleted and the WAIT state is maintained.

The function calls used in Table 6.1 are more complex. The following provides a short description of those calls:



	Current State	Event	Action	Subsequent State
1	IDLE	DATA.request	Save SDU Start DELAY_TIMER $count = 0$	SEND
2	SEND	ACKNOWLEDGE $count > 0$	Delete SDU Stop ACK_Timer Stop Ignore_Timer	IDLE
3		TRANSMISSION <b>indication</b> $delay\_ok$ <b>or</b> $count > 0$	Send SDU Start ACK_Timer Start Ignore_Timer $count = count + 1$	WAIT
4		TRANSMISSION <b>indication</b> $NOT\ delay\_ok$ $count == 0$	Delete SDU Start Refresh	IDLE
5		<b>DELAY_TIMER</b> <b>expired</b> $count > 0$	Delete SDU	WAIT
6		<b>DELAY_TIMER</b> <b>expired</b> $count == 0$	Delete SDU Start Refresh	IDLE
7		WAIT	ACKNOWLEDGE	Delete SDU Stop ACK_Timer Stop DELAY_Timer Reset_Delayed
8	REJECT $Not\ Ignore$ $SDU \neq NIL$		Stop ACK_Timer	SEND
9	REJECT $Not\ Ignore$ $SDU == NIL$		Reset_Poll Set_Delayed	WAIT
10	<b>ACK_Timer</b> <b>expired</b> $count < N2$		Start ACK_Timer $count = count + 1$ Set_Poll	WAIT
11	<b>ACK_Timer</b> <b>expired</b> $count \geq N2$		Delete SDU Stop DELAY_Timer Trigger Reset	IDLE
12	<b>DELAY_TIMER</b> <b>expired</b>		Delete SDU	WAIT

**Table 6.1:** State Transitions of the **Send\_Data\_Object**

**Stop ACK\_Timer** The Acknowledgement Timer is implemented into **Send\_Data** a single time for all **Send\_Data\_Objects**. In this case, stopping means that the timer is stopped if that object is the only one running the timer. In other words, **Send\_Data** determines whether the Acknowledgement Timer is stopped. A **Reset\_Poll** is also run.

**Start ACK\_Timer** The Acknowledgement Timer is only restarted if it is not yet running.

**Start Refresh** Firstly, it is important to switch to an IDLE state before starting the Refresh. The Refresh is executed by **Send\_Data** and updates the transmission window after a cell has been discarded before the first transmission. All subsequent cells are moved one slot up in the transmission window, so that the sequence is closed. Further Refreshes can be triggered during this process, thereby creating a recursion.

**Set\_Delayed Send\_Data** is shown that the cells were discarded and that the receiver needs to be informed thereof.

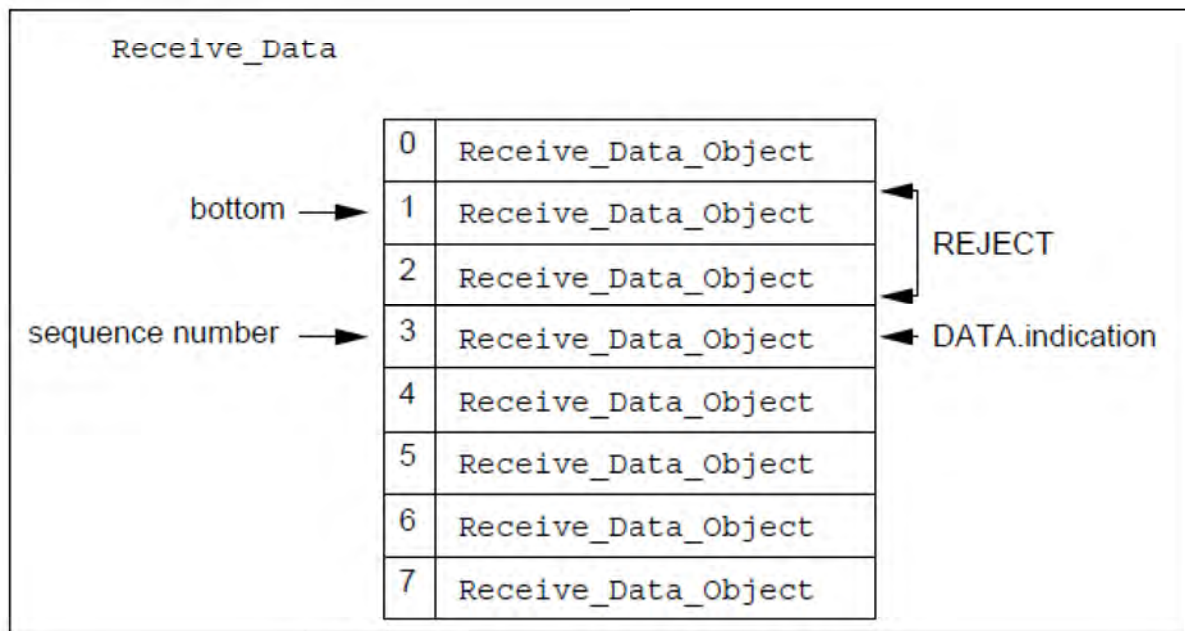
**Reset\_Delayed Send\_Data** is informed that a Delay PDU no longer needs to be sent for the cells.

**Set\_Poll Send\_Data** is ordered to set the Poll Bit if it has not already been set.

**Reset\_Poll** Polling is no longer necessary due to that cell.

**Trigger Reset** A Reset of the VC being run by the **Connection\_Handler** is triggered.

### 6.3 Receive\_Data



**Figure 6.5:** Events when a Frame Is Received

**Receive\_Data** manages the receiver window of the ASR ARQ protocol. A **Receive\_Data\_Object** is set up for each window slot and the object manages the slot independently. **Receive\_Data** can send events to the **Receive\_Data\_Object** without having to observe their status, as the objects independently recognize whether an event is meaningful for them. This has the advantage that the same method can be used every time a frame is received, regardless of the internal state. **Receive\_Data** has two pointers for managing the receiver window:

**bottom** always points to the lowest window position. Thus, it always points to the oldest cell that has not yet been received.

**reject\_number** points to the cell that should be requested next using an SREJ.

Figure 6.5 demonstrates the receiver mechanism. The figure shows a **Receive\_Data** instance with eight **Receive\_Data\_Object** instances. The maximum window size is  $n = 4$ . **Receive\_Data** then receives an *RR PDU* with sequence number 3. Slots from *sequence number* – 1 to *bottom* are then rejected, in this case that would be slots 2 and 1 (in that order). Next, the cell in slot 3 is saved. **Receive\_Data\_Object** decides how the rejection is handled.

The *bottom* and *reject\_number* pointers now need to be updated. The *bottom* pointer sends an *ACKNOWLEDGE* to the **Receive\_Data\_Objects** of the *bottom* and increases the *bottom* as long as the objects have the state RECEIVED. The *reject\_number* pointer is only changed if the number and position of the missing frames changes. This is shown by the individual **Receive\_Data\_Objects**. In addition to these tasks, the **Receive\_Data** is also responsible for managing the Reject Timer.

### 6.3.1 Receive\_Data\_Object

**Receive\_Data\_Object** manages a receiver window slot. It is implemented as a finite state machine and can assume the states shown in Figure 6.6:

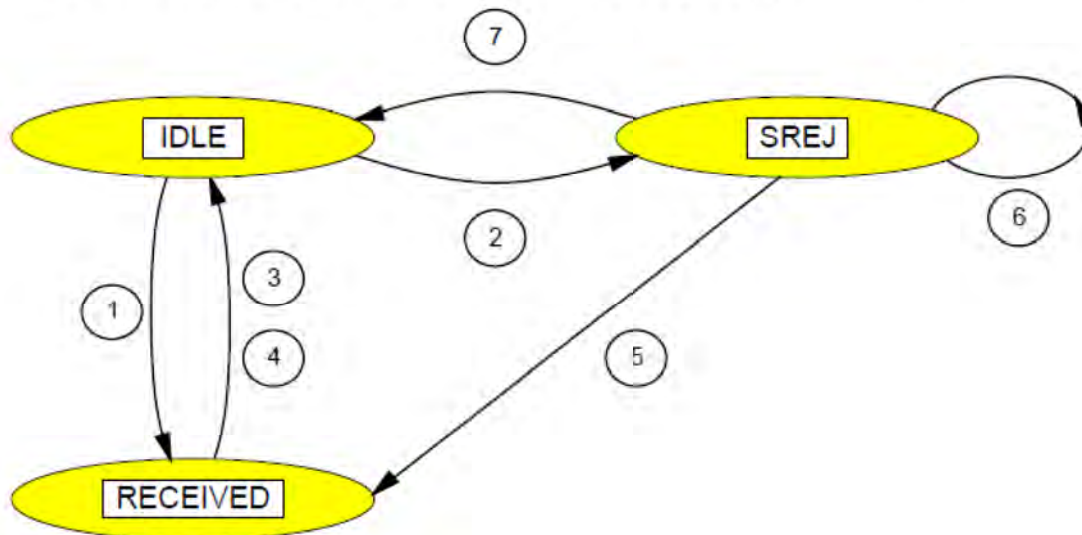


Figure 6.6: States of the **Receive\_Data\_Object**

#### IDLE

The **Receive\_Data\_Object** is in an IDLE state following the initialization. If a *DATA.indication* is received, the SDU is saved, the *FORWARD\_Timer* is started and the state is changed to RECEIVED (1). If an SDU is missing from the receiver sequence (*REJECT*), the *REJECT\_Timer* is started and the reject count is set to *reject\_count* = 0. **Receive\_Data\_Object** transitions into the SREJ state (2).

	Current State	Event	Action	Subsequent State
1	IDLE	<b>DATA indication</b>	Save SDU Start FORWARD_Timer	RECEIVED
2		<b>REJECT</b>	Start REJECT_Timer <i>reject_count = 0</i>	SREJ
3	RECEIVED	<b>ACKNOWLEDGE</b>	Forward SDU Stop FORWARD_Timer	IDLE
4		<b>FORWARD_Timer expired</b>	Start Forwarding	IDLE
5	SREJ	<b>DATA indication</b>	Save SDU Stop REJECT_Timer Start FORWARD_Timer	RECEIVED
6		<b>REJECT_TIMER expired</b> <i>reject_count &lt; N2</i>	<i>reject_count ++</i> Start REJECT_Timer	SREJ
7		<b>REJECT_Timer expired</b> <i>reject_count ≥ N2</i>	Trigger Reset	IDLE

**Table 6.2:** State Transitions of the **Receive\_Data\_Object**

### RECEIVED

The object remains in a RECEIVED state while waiting for the missing SDUs from other **Receive\_Data\_Objects** to arrive. An **ACKNOWLEDGE** is issued when the SDUs are received. The SDU is forwarded to the upper layer, the FORWARD\_Timer is stopped and the object returns to an IDLE state (3). If the FORWARD\_Timer runs out, the forwarding is started and the object transitions to an IDLE status (4).

### SREJ

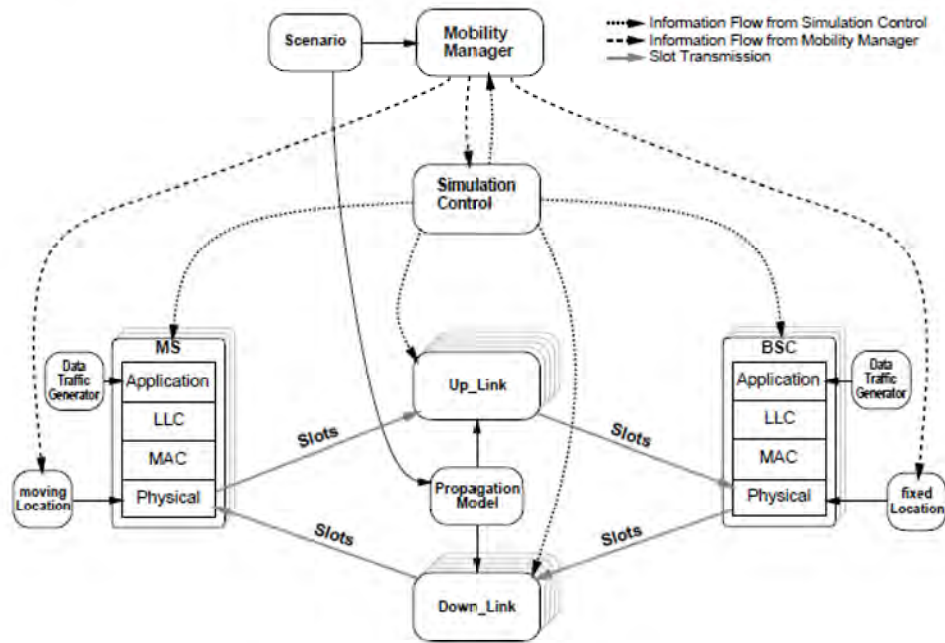
The object remains in an SREJ state while waiting for a missing SDU. If the SDU arrives (*DATA.indication*), the SDU is saved, the REJECT\_Timer is stopped, the FORWARD\_Timer is started and the object moves into a RECEIVED status (5). If the REJECT\_Timer runs out, the number of new requests should be noted. If the *reject\_count < N2*, the REJECT\_Timer is restarted, the *reject\_count* is increased by one and the wait for the missing SDU is continued (SREJ state) (6). If the *count > N2*, a local reset is triggered and the object returns to the IDLE state (7).

Function calls used in Table 6.2 are more complex. The information on the **Acknowledgement\_Timer** is also true for **Start Reject\_Timer** and **Stop Reject\_Timer** (chapter 6.2.1). The same applies for **Trigger Reset**. In case of **Start Forwarding**, **Receive\_Data** is informed that it should no longer wait for the missing SDUs. **Receive\_Data** then forwards the SDUs in the temporary archive to the upper layer and shifts its window accordingly.

## Implementing the ASR ARQ Protocol

### 7.1 Overview

The simulator is implemented in the object-based programming language C++. The simulator follows an event-driven simulation principle. The classes required for doing so are provided in a class library *Communication Networks Class Library CNCL* developed by the Chair of the Communication Network Department.



**Figure 7.1:** The Simulator Modules

The layout of the simulator and the flow of information between the individual modules are shown in Figure 7.1. The simulator links communication objects to their environment. The *Base Station Controller* and the mobile stations are the communication objects. The environment is modeled by the mobility of the communication objects and the physical propagation of the radio signals. A uniform scenario that has equal influence on the physical radio propagation and the mobility describes the simulation environment.

The modules containing the mobility, or the radio propagation, are separated structurally from the rest of the system through their object-based implementation. The description approaches, which in and of themselves are very simple, can easily be swapped for different, much more complex model descriptions later on.

A comparable implementation approach was followed for the layout of the protocol stack in the *Base Station Controllers* and mobile stations. The object-based classification of the different layers, achieved through the definition of clear interfaces, promotes the interchangeability and reusability of the specified protocols. The protocol stacks are structured pursuant to the ISO/OSI reference model. Classes from the SIMCO library are used in the C++ implementation for communication between the layers. Figure 7.1 shows the most important objects, these are:

**Simulation\_Control** The simulation is controlled by the central control unit **Sim\_Control**. It is responsible for generating and deleting the *Base Station Controller* **BSC** and mobile stations **MS**. In turn, **BSC** and **MS** generate the individual protocol layers of the simulator.

**Mobility\_Manager** The mobility manager enables both centrally- and locally-managed mobility. Although the functionality is allocated to both dynamically-generated **Location Objects** and **Location Objects** assigned to the communication objects while doing so, there is also a central control unit. Depending on which mobility model is implemented, the central unit has more functionalities than the dynamic objects.

**MS and BSC** The individual layers of the simulator are instantiated and deleted in classes **MS** and **BSC**. Each object of the above-mentioned classes represents a mobile station or a *Base Station Controller*.<sup>10</sup>

<b>.Application_Type</b>	<b>Test_LLC</b>	Testing of the LLC layer
	<b>stat_APP</b>	Evaluation of the performance of the LLC layer
<b>.LLC_Type</b>	<b>LLC</b>	Splitting into <b>.Upper_LLC</b> and <b>.Lower_LLC</b>
	<b>LLC_stat_MAC</b>	Evaluation of the performance of the MAC layer
<b>.Upper_LLC_Type</b>	<b>Upper_LLC</b>	Full functionality
	<b>Stat_LLC_Lower</b>	Evaluation of the performance of <b>Lower_LLC</b>
<b>.Lower_LLC_Type</b>	<b>Lower_LLC</b>	Full functionality
	<b>Test_Upper_LLC</b>	Testing of <b>Upper_LLC</b>
<b>.MAC_Type</b>	<b>MAC_DSA</b>	DSA protocol
	<b>MAC_DSA++</b>	DSA protocol++
	<b>MAC_TEST_LLC</b>	Testing of <b>Lower_LLC</b> and <b>Upper_LLC</b>

**Table 7.1:** Protocol Variants of the Different Layers in **.sim\_defaults**

The protocol stack is constructed in different layers with different fields of responsibility pursuant to the ISO/OSI reference model. Since there are different ways to fulfill the tasks of a layer or sub-layer, it needs to be possible to swap a protocol from a layer with another protocol from the same layer in order to simulate different protocols. Different layer types can be selected when a simulation is started. This task is realized in the objects of the **MS** and **BSC** classes using a **.sim\_defaults** file (chapter 7.5). The layer types registered in the **.sim\_defaults** file are instantiated, initialized and then started. The individual layers are deleted at the end of the

program run in the destructors found in the **MS** and **BSC** classes. Table 7.1 shows the wide range of different protocols and their meaning. The LLC layer is split into two again for LLC type *LLC* and these can be configured independent of one another.

The **stat\_APP**, the sources for load generation and the **MAC\_Test\_LLC** are particularly important for the simulative evaluation of the ASR ARQ protocol.

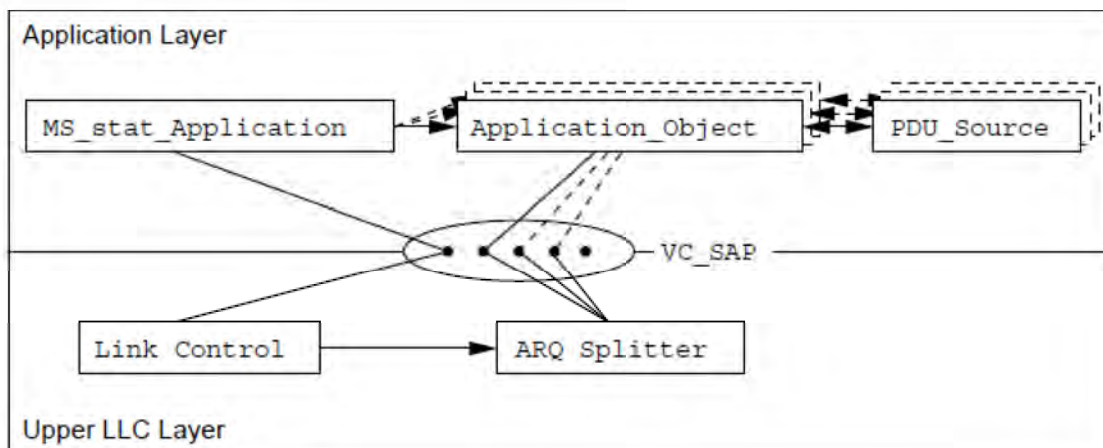
## 7.2 Application Layer stat\_APP

The *Application Layer* represents the layers above the LLC layer. It is responsible for generating communication loads for specified virtual channels. The *Application Layer* must have the following properties in order to ensure that the simulated scenarios are as realistic as possible:

- Communication via the service access point in the LLC using *Service Primitives*
- Connection of the application instances with negotiated terminals
- Reasonable reaction to connection establishment, termination and reset requests
- Generation of use data based on different source models
- Requirements for specified virtual channels
- Flexibility in regard to station and channel quantities and source types. Specifically, it should be possible to generate different stations with different channel characteristics.

These requirements can be split into the generation of use data in the *sources* (chapter 7.3) and the management of the virtual channels.

### Mobile Station



**Figure 7.2:** Application Layer in the Mobile Stations

Figure 7.2 demonstrates that all communication between the Application Layer and the Upper LLC Layer is realized through the **VC-SAP** (Virtual Channel Service Access Point). Table 7.2 provides information on the usable SPs and their meaning.

An **MS\_stat\_Application** object that initiates the construction of all specified virtual channels is generated for each mobile station. The mobile station can only initiate the construction of one channel due to the *Resource Management* in the lower LLC layer [6]. Accordingly, **MS\_stat\_Application** must also trigger the construction of the channels that only

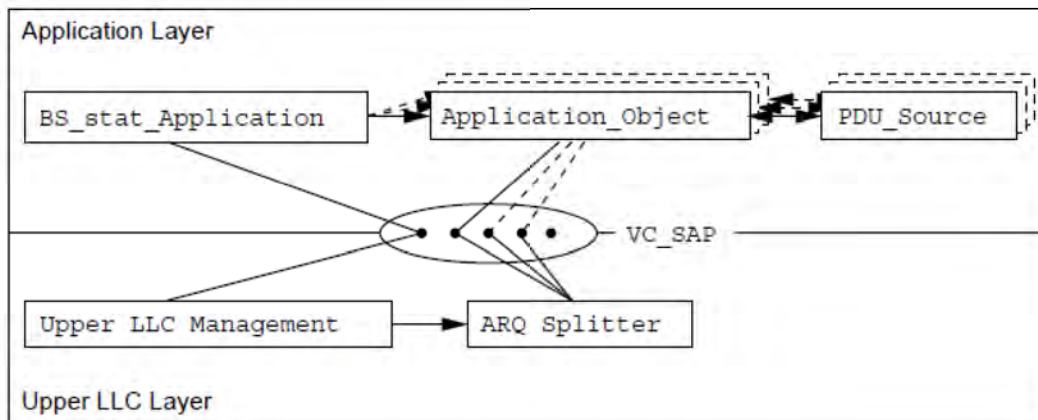
send information on the downlink. In order to construct a channel, the Application Layer must be aware of the *Mean Rate*, *Peak Rate* and *maximum delay* of both the uplink and downlink. This data can then be

Service Primitive	Generating Layer	Significance
L_Connect_Request	Application	Connection request
L_Connect_Indication	LLC	Display of a connection request
L_Connect_Response	Application	Confirmation of a connection request
L_Connect_Confirm	LLC	Confirmation of a connection request
L_Disconnect_Request	Application	Connection release
L_Disconnect_Indication	LLC	Display of a connection release
L_Reset_Request	Application	Release request
L_Reset_Indication	LLC	Display of a reset request
L_Reset_Response	Application	Confirmation of a reset request
L_Reset_Confirm	LLC	Confirmation of a reset request
L_Data_Request	Application	Transfer of use data for transmission
L_Data_Indication	LLC	Transfer of received use data

**Table 7.2:** Service Primitive of the Application Layer

used to send an **L\_Connect\_Request** SP to the **Link Control**. The establishment of the connection is negotiated using connection terminal **0**. In the LLC layer, the **Link Control** that triggers the construction of the corresponding ARQ instances is also connected here (chapter 6).

#### Base Station



**Image 7.3:** Application Layer in the Base Station

Once an **L\_Connect\_Indication** SP with a positive answer or an **L\_Connect\_Confirm** SP has been received, **MS\_stat\_Application** sets up an **Application\_Object** that is connected to the transmitted CEP in the **VC\_SAP**. This object constructs the specified source and begins generating use data. Received use data is evaluated statistically in the Application Layer and then destroyed. Once the VC has been constructed, the communication for the channel runs exclusively between the **Application\_Object** and the **ARQ Splitter**.

A **BS\_stat\_Application** object is generated in each base station (Figure 7.3), that reacts to connection establishment and termination requests accordingly. In contrast to the mobile station, **BS\_stat\_Application** can only initiate connection releases. In the base station, its contact



partner is the **Upper LLC Management**, which forwards SPs to the responsible **Link Control** objects. In the base station, only the **Upper LLC Management** has enough information on the responsible Upper LLC Block.

Whereas the ARQ abbreviated address of the virtual channel is used as a CEP address in the mobile station, an available connection terminal address has to be determined by the *ARQ Splitter* in the base station and shared with the **BS\_stat\_Application**. This is necessary as ARQ abbreviated addresses are only unique in the base station when used in connection with the mobile station ID, i.e. different mobile stations can use the same ARQ addresses.

### 7.3 Sources

The following source types are implemented:

- Poisson
- LAN
- Video
- CBR (Constant **Bit Rate**)

Based on their characteristics, the sources calculate the interim arrival time of the next PDU and trigger the generation of the PDU based on the calculated time.

#### 7.3.1 Poisson Source

With this type of source, the interim arrival times can be allocated negatively exponentially. The result is that the individual times are fully uncorrelated, i.e. the interim arrive time of the next cell is independent of the preceding cell. This is referred to as a memoryless source. Thus, a *Peak Rate* cannot be specified. The mean data rate uniquely describes the Poisson source.

Even if it can be assumed that the data arrival in MBS will *not* be negatively exponentially allocated, there are still very general demands on the protocol since the interim arrival times are uncorrelated.

#### 7.3.2 LAN Source

The LAN source models the data communication in a **Local Area Network**. The following data is based on measurements in existing networks [35]. There are four different types of service:

**Terminal Input** refers to the user input in a terminal.

**Terminal Response** is a computer's answer to a terminal request.

**File Transfer** describes the transfer of entire files.

**Paging** is the swapping of main memory pages for workstations without their own mass memory.

There are also five different types of stations:

**Workstation** describes a high-performance personal computer, which generates Terminal Response and File Transfer services.

**Diskless Workstation** describes a high-performance personal computer that does not have its own hard drive. It generates Terminal Input and Paging services.

**Department Host** is used in the scenario as a file server and generates Terminal Response, File Transfer and Paging services.

**Gateway** provides the connection to other computer networks. Terminal Responses and File Transfers are generated from the gateway.

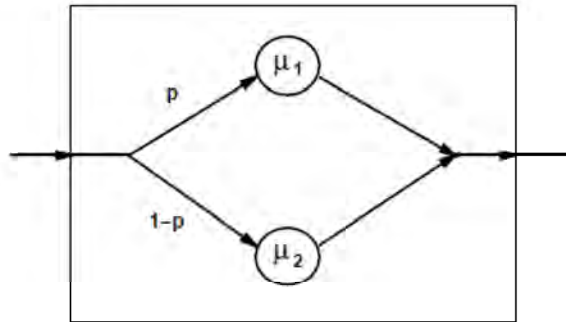
**Central Host** is a high-performance host system. It generates Terminal Response and File Transfer services.

These station types can generate the different services with different package lengths. The package length is distributed hyperexponentially based on an  $H_2$  distribution with the following mean values and relative variances:

	Terminal Input	Terminal Response	File Transfer	Paging
Mean value $\bar{x}$ [Bytes]	12	180	40000	8192
Relative variance $c$	1.3	6.1	6.0	0

**Table 7.3:** Mean Values and Variance by Service Class

The values in Table 7.3 produce the following values for the values in the  $H_2$  distribution model (Figure 7.4):



$$p = \frac{1}{2} + \frac{1}{2} \cdot \sqrt{1 - \frac{1}{1+c^2}} \quad (7.1)$$

$$\mu_1 = \frac{2 \cdot p}{\bar{x}} \quad (7.2)$$

$$\mu_2 = \frac{2 \cdot (1-p)}{\bar{x}} \quad (7.3)$$

**Figure 7.4:**  $H_2$  Distribution Model

Equations (7.1), (7.2) and (7.3) produce the following equation for the package length:

$$P[X \leq x] = 1 - p \cdot e^{-\mu_1 x} - (1-p) \cdot e^{-\mu_2 x} \quad (7.4)$$

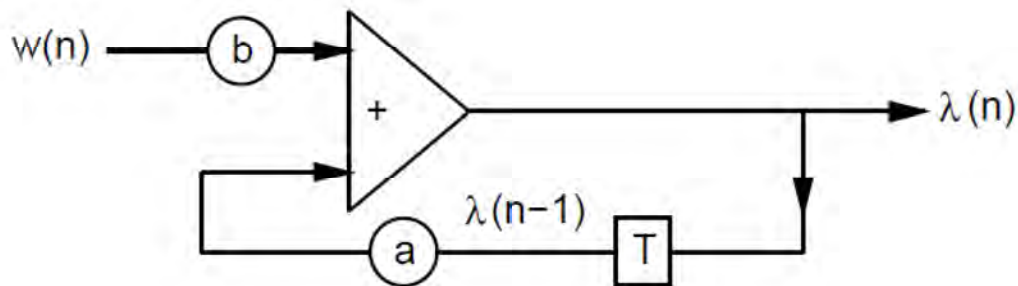
Table 7.4 shows the mean value of the number of services per hour by station type. The interim arrival time between two services of a class is distributed negatively exponentially. The cells of a package are generated at a rate 1, i.e. one in each slot. The entire mean data rate is transferred to the source and the individual data rates are adjusted accordingly.

### 7.3.3 Video Source

This source simulates high-quality video-telephony and is based on suggestions from [4].

Station Type	Terminal Input	Terminal Response	File Transfer	Paging
Workstation	0	2500	60	0
Diskless	750	0	0	360
Department Host	0	31500	296	1800
Gateway	0	21250	128	0
Central Host	0	55000	832	0

**Table 7.4:** Mean Value of the Number of Services per Hour



**Figure 7.5:** Autoregressive Process

A 1<sup>st</sup> order autoregressive process is used to model the number of packages per image (Figure 7.5). The number of bits per pixel for the n<sup>th</sup> image can be determined by:

$$\Lambda(n) = \max \{ a \cdot \lambda(n - 1) + b \cdot w(n), 0 \} \tag{7.5}$$

where  $\lambda(0) = 0.52$ ,  $a = 0.878$ ,  $b = 0.111$  and  $w(n)$  is a Gaussian distribution with  $\eta = 0.572$  and  $\sigma^2 = 0.0536$ .

With 30 images per second and 250,000 pixel resolution per image, this source produces an average of 170 cells per image, which corresponds to 1.9 Mbit/s. The interval between two images is 1548 slots. This source, in turn, is scaled to a total mean data rate. All cells of an image are virtually generated at the same time.

### 7.3.4 CBR Source

The interim arrival time for this is given as a constant using:

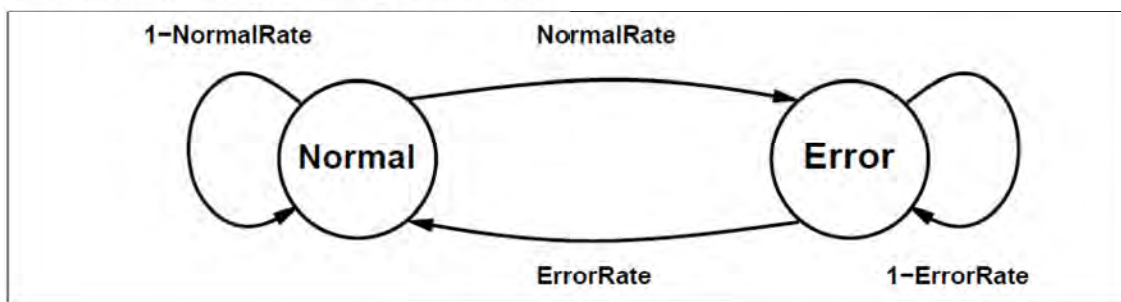
$$l = \frac{1}{\text{mean rate}} \lfloor T_{slot} \rfloor \tag{7.6}$$

It is deterministically predictable. It thus places a lower demand on the ASR ARQ protocol and is only used in source combination together with the other sources presented here.

#### 7.4 The MAC\_Test\_LLC MAC Layer

The **MAC\_Test\_LLC** provides all of the lower layers for the LLC layer, in other words the MAC and the physical layer.

The influence of a specific MAC protocol was eliminated in order to determine the effects of different aspects of the protocol. Instead, the MAC layer is modeled by an ideal scheduler using the G/D/1/FCFS/RU-NONPRE strategy. When doing so, it is assumed that the scheduler has ideal knowledge of the capacity requirements. Short supervisory frames cannot be transmitted and acknowledgments are only transmitted when piggybacked to information frames, possibly with an empty information field, if necessary. The measured delays are thus always lower than in a real system. The ratio between allocations of transmission delays, which are necessary for evaluating protocol options, is not affected. The channel model corresponds to an indoor scenario with negligible signal runtimes.



**Figure 7.6:** Gilbert Model

A Gilbert model is used to recreate the bit error rates with two states (Figure 7.6). There are no bit errors in the **Normal** state. A bit error occurs in the **Error** state that leads to the frame loss rate defined in the **PDUErrorRate**. The mean error cluster length is defined in **StateErrorRate** as  $\frac{1}{T_{slot}}$ . The mean error rate can be determined as follows when the mean length of error-free periods is specified in **StateNormalRate**:

$$\text{mean error rate} = \frac{\text{StateNormalRate}}{\text{StateErrorRate}} \cdot \text{PDU Error Rate} \quad (7.7)$$

This model allows the bit error that is typical for a broadband channel to be parameterized [7, 33] (mean error cluster length =  $50 T_{slot}$ , bit error rate in a poor state =  $10^{-3}$ ; mean bit error rate =  $10^{-4}$ ). Due to the full duplex transmission to two different FDM channels, it is assumed that the uplink and downlink are uncorrelated. Further, it is assumed that all errors in that layer can be identified; meaning that the bit errors manifest themselves when the entire frame is discarded and no erroneous frames are forwarded to the LLC layer.

To summarize, the **MAC\_Test\_LLC** has the following functionality:

- Capacity is provided to the station with the highest priority.
- There is a consistent transmission delay of  $0.9 T_{slot}$ .
- Erroneous frames are discarded in full.
- Parameters can be defined for the error rate and its correlation.

## 7.5 The .sim\_defaults Configuration File

<b>Sim_Length:</b>	Simulation duration, specified in “slots”
<b>Sim_No:</b>	Used to identify and differentiation between the output files
<b>Output.directory:</b>	File to which the output files are saved
<b>Parameters</b>	
<b>.LLC</b>	
<b>.ConSetup_TimerDelay:</b>	Time left until the timer in the <b>ConnectionHandler</b> runs out
<b>.N2:</b>	Maximum number of transmission attempts in the <b>ConnectionHandler</b>
<b>.CheckTime:</b>	Time left until sources will be started in <i>BSC</i>
<b>.GIST_Delay:</b>	Time left until GIST is started
<b>.ouput_interval:</b>	Time between two file outputs
<b>.max_ms_in_cell:</b>	Maximum number of mobile stations that can accept an MAC instance in the <i>BSC</i>
<b>.max_vc_in_ms:</b>	Number of virtual channels that the LLC layer can accept
<b>.max_ms_in_bs:</b>	Number of mobile stations that the LLC layer can accept
<b>.MACMAXCEP:</b>	Number of connection terminals in the mobile station
<b>.stat_Application:</b>	
<b>.NoPDUs:</b>	Number of PDUs generated per VC
<b>.Nr_Uplink_VC:</b>	Maximum number of VCs in an MS
<b>LLC</b>	
<b>.VC_Send_Prio_method:</b>	Algorithm for <b>ARQ Send Priority</b>
<b>.VC_Receive_Prio_method:</b>	Algorithm for <b>ARQ Receive Priority</b>
<b>.MaxQueueLength:</b>	Maximum number of queue slots
<b>MAC.Test_LLC</b>	
<b>.PDUErrrorRate:</b>	Error rate in the Error state
<b>.StateErrorRate:</b>	Rate for exiting Error
<b>.StateNormalRate:</b>	Rate for exiting Normal
<b>Mobility.MS_Density:</b>	Number of mobile stations
<b>debug_LL_up:</b>	Debug information for the upper LLC layer

**Table 7.5:** Parameters in the .sim\_defaults Configuration File

The .sim\_defaults file contains a series of simulation parameters. They are imported and processed at the beginning of the simulation. This allows different simulations with different parameters and different layer protocols to be run, without having to recompile. Table 7.5 shows the

parameters that are most important in the upper LLC layer.

<b>Uplink_VC</b>	
<b>.VCx</b>	<i>x = virtual channel number</i>
<b>.MobId:</b>	Mobile station ID
<b>.VcId:</b>	Virtual channel ID
<b>.Windowsize:</b>	Maximum window size in the sender and receiver
<b>.max_sequence_number:</b>	Modulus for coding the sequence numbers
<b>.Ack_Threshold:</b>	Priority threshold, empty frames will be sent once it has been reached
<b>.ARQ_TimerDelay:</b>	Acknowledgement Timer delay
<b>.RejectTimer_used:</b>	Use Reject Timer
<b>.Reject_TimerDelay:</b>	Reject Timer delay
<b>.PTimer_used:</b>	Use Poll Bit Timer
<b>.P_Timer_Delay:</b>	Poll Bit Timer delay
<b>.DelayTimer_used</b>	Use Delay Timer and Delay PDU
<b>.ForwardTimer_used:</b>	Use Forward Timer to cancel the re-sequencing in the receiver
<b>.IgnoreTimer_used:</b>	Use Ignore Timer
<b>.N2:</b>	Number of transmission attempts until a reset
<b>.Mean_Rate</b>	Mean data rate of the source
<b>.Source_Type</b>	Source type
<b>.Delay:</b>	Maximum delay for ATM cells
<b>.LRE_used:</b>	Use LRE for static evaluation

**Table 7.6:** Parameters in the **.uplink\_vc\_llc** Configuration File

To maintain a clear overview, the parameters for specifying the individual virtual channels were moved to the **.uplink\_vc\_llc** and **.downlink\_vc\_llc** files. Each of the parameters listed in Table 7.6 can be set for each virtual channel in these files. In addition, the number of virtual channels being used must also be specified in **.Uplink\_VC.Nr\_of\_VCs\_used**. Parameters are initiated using **Downlink\_VC** instead of **Uplink\_VC** in **.downlink\_vc\_llc**. Both of these files are evaluated and managed by **VC\_Base**. This allows the corresponding values to be queried when the **MobId** and **VcId** are specified.

## 7.6 The Graphical Debugger

Protocol processes in the LLC layer can be displayed using the GIST program (*Graphical Interactive Simulation result Tool*). GIST makes it possible to record the temporal changes in the simulation states; the changes can then be rewound and replayed in the same manner as a video recorder to demonstrate the increments of the protocol, i.e. to allow errors to be detected.

The **LLC-Graphic** interface was developed in [10] specifically for visualizing the ARQ protocol in the LLC layer; the interface can display all processes that are important for the protocol. A list of the function defined in the interface can be found in [34].

Figure 7.7 shows an example of the screen display in GIST.

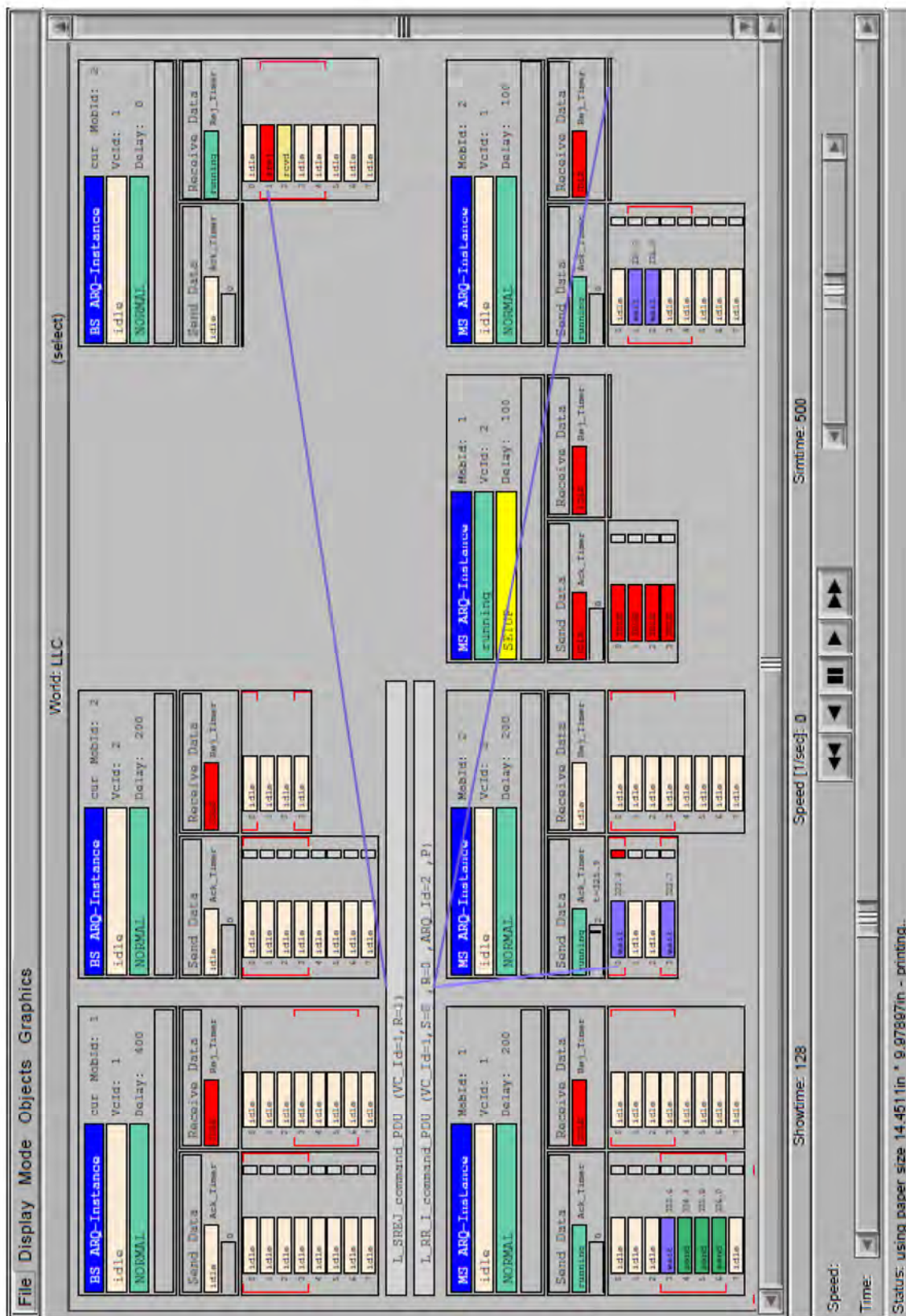
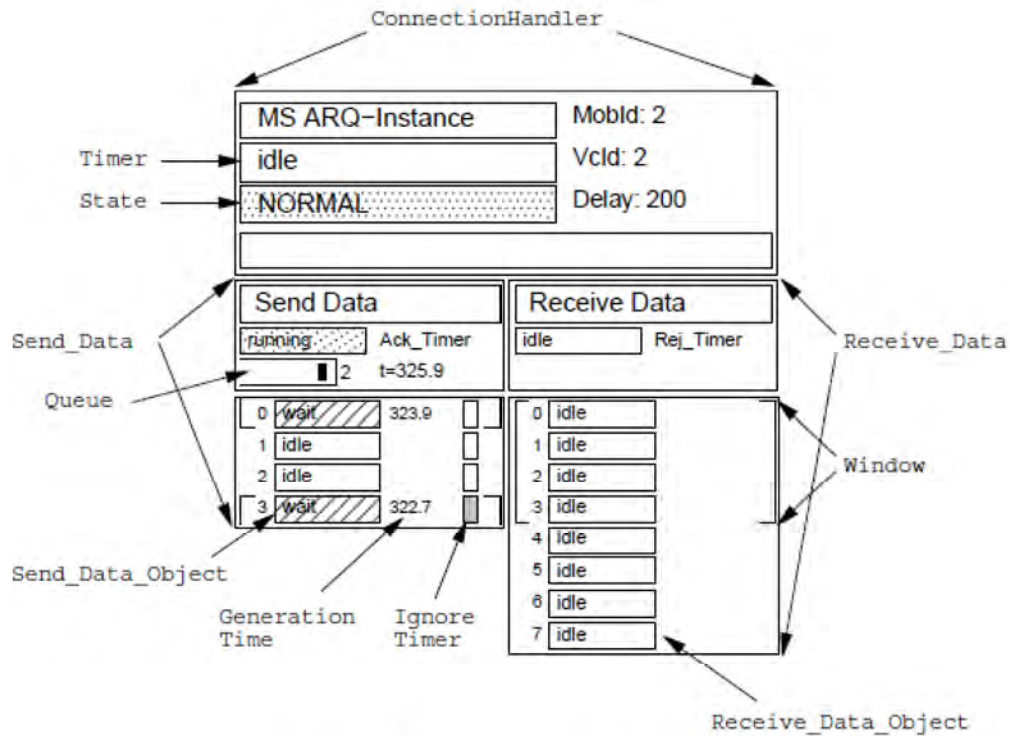


Figure 7.7: Screen Display in GIST



**Figure 7.8:** Elements in GIST

The following elements can be displayed in GIST:

- The **ConnectionHandler** and its state and timer
- **Send\_Data** with the position of the transmission window, with the state of the queue and the state of the Acknowledgement Timer
- **Send\_Data\_Object** with its state, the date/time a saved cell was created and the state of the Ignore Timer
- **Receive\_Data** with the position of the receiver window and the state of the Reject Timer
- **Receive\_Data\_Object** with its state and the date/time a saved cell was created
- Type and origin of the transmitted ARQ frames

Please see [34] for the implementation details.



## Simulation Results

### 8.1 Implementation

The goal of the simulations is to evaluate the performance of the ASR ARQ compared to traditional ARQ protocols and to an idealized ATM Multiplexer. Accordingly, the relative performance is one of the main evaluation criteria. Further, the only statements that can be reasonably made are relative statements, since simulations are only possible with an idealized MAC layer. However, this also achieves a situation where any aspects causing the measurement values to vary are located exclusively within the LLC layer.

#### 8.1.1 Measurements and Performance Parameters

ITU-T.356 provides the following performance parameters for ATM [18]:

CER Cell Error Ratio

$$CER = \frac{\text{total errored cells}}{\text{total successfully transferred} + \text{total errored cells}}$$

CLR Cell Loss Ratio

$$CLR = \frac{\text{total lost cells}}{\text{total transmitted cells}}$$

CMR Cell Misinsertion Rate

$$CMR = \frac{\text{total number of misinserted cells in time interval}}{\text{time interval duration}}$$

SECBR Severely Errored Cell Block Ratio

$$SECBR = \frac{\text{total severely errored cell blocks}}{\text{total cell blocks}}$$

CTD Cell Transfer Delay

$$CTD = \text{time between arrival in sender and departure in receiver}$$

CER, CMR and SECBR cannot be measured as the MBS simulator does not have a realistic model for generating erroneous cells.

The values that are measured, besides the CTD, include:

**discarded ratio**

$$\frac{\text{number of discarded cells}}{\text{number of generated cells}}$$

**throughput**

$$\frac{\text{number of forwarded cells}}{\text{number of PDUs sent}}$$

**multitransmission ratio**

$$\frac{\text{number of information PDUs received} - \text{number of forwarded cells}}{\text{number of forwarded cells}}$$

The *CLR* can be determined from the *discard ratio* and the distribution function of the cell delays. *CLR* and *CTD* are the QoS parameters in the simulations and define the conditions the VC must meet. The *CTD* is used to determine the *MCTD* (Mean Cell Transfer Delay) as an arithmetic average and the *CDV* (Cell Delay Variation).

**8.1.2 Static Significance**

There is a finite sequence of measurements for each random variable to be determined during the simulation that must be evaluated using a static approach. Typical conditions in the performed simulations are:

- The random process is stationary
- Measured values are correlated among one another
- The random process type is unknown
- The number of measurements in a range from  $10^6$ - $10^7$

Due to these conditions, the static evaluation process should make the following statements:

- Empiric distribution function
- Empiric moments
- Correlation statements
- Object error measurement

Using a conventional approach based on batch means, the random sequence is split into equal partial sequences, which are basically regarded as independent. This creates an implicit requirement for the total random values to have a normal distribution. However, this is only the case with uncorrelated random sequences. Even the confidence intervals that were derived as an error measurement are mainly based on assumptions, meaning their applicability is conditional.

The use of the Bayes-Laplace statistic solves this problem [11]. It is used advantageously in the LRE (**L**imited **R**elative **E**rror) algorithm. The LRE allows the simulation duration to be monitored based on the relative error. Please see [11] for details on the implementation of the Bayes-Laplace statistic and the different LRE algorithms.

The LRE algorithm implemented in the CNCL library was used to monitor the simulation duration and the relative error. The relative error in the following simulations is less than 1% in the displayed range.

### 8.1.3 Simulation Parameters of the ASR ARQ

The following causal relationships exist between the individual parameters shown Table 8.1: the value of the *modulus* is a system parameter that is defined when the system is drafted and cannot be changed dynamically. In addition to the *modulus*, the maximum window size is also a fixed value (chapter 5.1.3). Thus, the values of *modulus* = 8 and *window size* = 4 are used in the following.

Class	Name	Significance
<b>Timer</b>	Acknowledgement	Waits for acknowledgement
	Reject	Waits for frame for which a new request was made
	Poll	Waits for answer to a poll
	Delay	Monitors remaining life cycle in the sender
	Forward	Monitors remaining life cycle in the receiver
	Ignore	Use of $T_{loop}$
<b>Window</b>	Modulus	Field dimension in PDU frame for SN, RN
	Window size	Maximum window size
<b>Control</b>	N2	Number of multitransmissions
<b>Priority Algorithm</b>	Send_Prio	Algorithm for determining the sending VC
	Receive_Prio	Algorithm for determining the acknowledging VC
	Ack_Threshold	Priority threshold

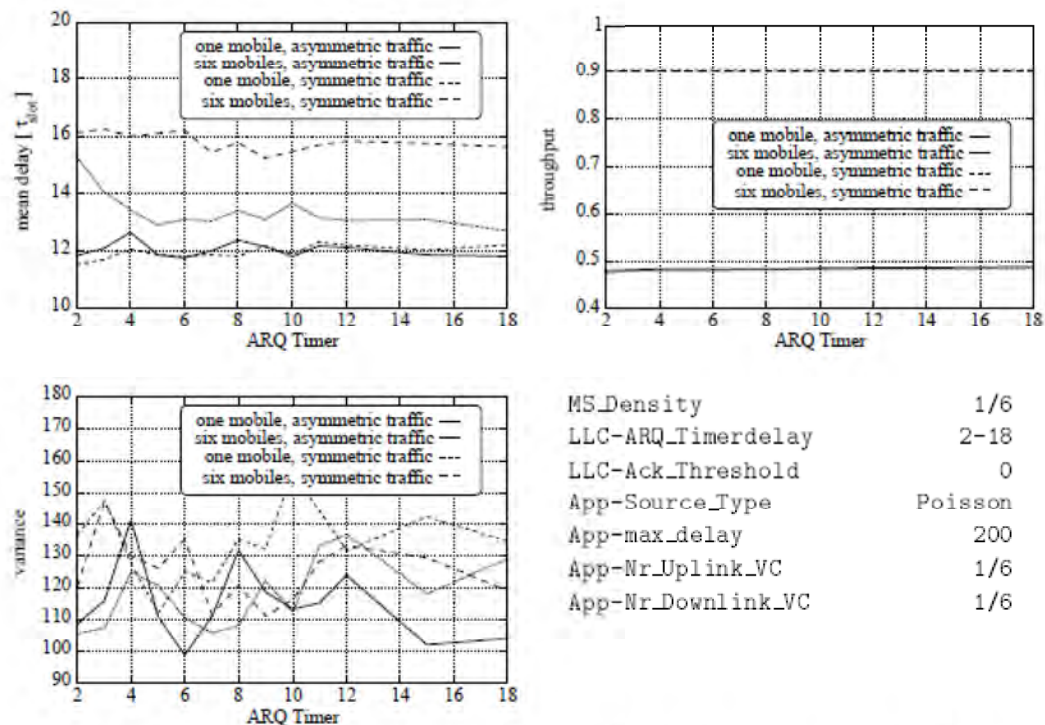
**Table 8.1:** Overview of the ASR ARQ Parameters

$N_2$  is always set in a manner that prevents resets. However, if a reset occurs anyway, it is very likely that an error in the protocol caused a deadlock (see chapter 5.2.4 for examples). Only **Short\_Remain** was used for *Send\_Prio* (see chapter 6). Thus, the variable simulation parameters that are left for ASR ARQ are the Timer, *Receive\_Prio* and the corresponding *Ack\_Threshold*.

The following will attempt to demonstrate the dependency between these parameters and on the window size.

## 8.2 Optimizing the ARQ Timer Delay and Ack\_Threshold Parameters

### 8.2.1 ARQ Timer Delay



**Figure 8.1:** Mean Delay, Variance and Throughput with Variation of ARQ Timer Delay

The first thing examined was what sort of influence the *ARQ Timer Delay* parameter had in the different scenarios. To do so, all of the other timers were turned off and the *oldest\_first* acknowledgement strategy was simulated without *Ack\_Threshold*. The entire traffic load in all simulations is 90%, i.e. with an uncorrelated frame error rate of 5% the channel has a high workload. Four different scenarios are considered:

**one mobile, asymmetric traffic** Six VCs, each with 15% load on the uplink, are operated in one mobile station.

**six mobiles, asymmetric traffic** One VC with 15% load on the uplink is operated in six different mobile stations.

**one mobile, symmetric traffic** Six VCs, each with 15% load on the uplink and downlink, are operated in one mobile station.

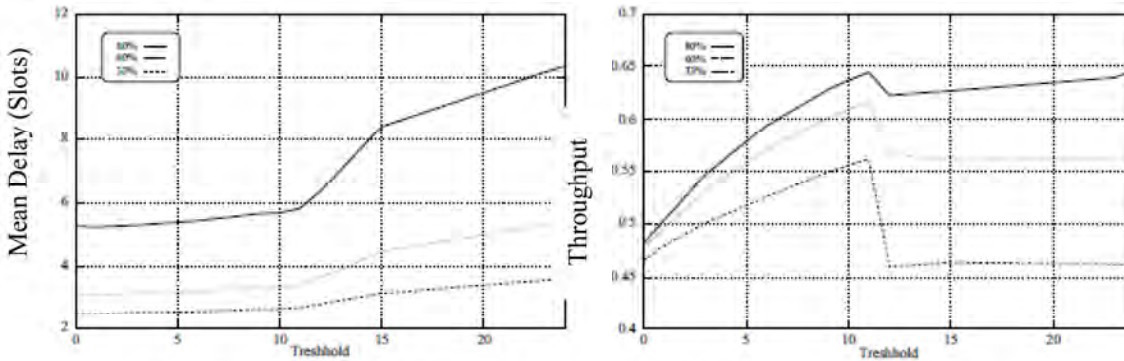
**six mobiles, symmetric traffic** One VC with 15% load on the uplink and downlink is operated in six different mobile stations.

The diagrams shown in Figure 8.1 lead to a single conclusion: the *ARQ Timer Delay* parameter does not have a decisive influence on the simulation, i.e. no system demonstrates the best way to select the delay. It merely shows that selecting a delay that is too short can lead to additional traffic in some scenarios.

The polling is obviously performed so seldom, due to the expiry of the *ARQ Timer*, i.e. the additional polling is transmitted piggyback, that it has no decisive influence. However, there is a dimensioning process for the *ARQ Timer Delay*, which experience has proven to be true and which I would like to share at this point:

$$(8.1) \quad ARQ\ Timer\ Delay = 2 \cdot \frac{0.9}{mean\ rate}$$

### 8.2.2 Ack\_Threshold



**Figure 8.2:** Mean Delay and Throughput when Traffic Is Unidirectional

The goal of the simulation is to examine the costs of bundling acknowledgements. Bundling should increase the throughput without causing a significant increase in the delay. Moreover, the dependency of **Ack\_Threshold** on the **ARQ Timer Delay** will also be examined.

MS_Density	6
LLC-ARQ_Timerdelay	12
LLC-Ack_Threshold	0-24
App-Source_Type	Poisson
App-max_delay	200
App-Nr_Uplink_VC	1
App-Nr_Downlink_VC	0

**Table 8.2:** Simulation Parameters

As shown in Table 8.2, a scenario with six mobile stations, each with one VC, is examined. Use data is transmitted exclusively over the uplink, so that the downlink is available for the acknowledgments. In order to compare them, the value *ARQ Timer Delay* = 12 is used and the **Acknowledge Threshold** varies in a range between 0-24. The total supply is 33%, 60% and 80% and is distributed equally among the stations. Poisson sources are used to generate the load.

Figure 8.2 reveals that the mean delay only shows a slight increase up to a threshold of 11, but the throughput increases considerable. At a threshold of 12 the throughput collapses and only increases when there is a higher load, but the mean delay increases noticeably.

When the **Acknowledge Threshold**  $\geq 12$ , the sender does not receive the acknowledgements in time and polls the receiver. The polling can be transmitted by piggyback more often when there is more traffic than when there is less. Thus, the collapse of the throughput is greater with a smaller load than with a greater load. With a smaller load, the polling completely destroys any advantages gained by bundling the acknowledgements.

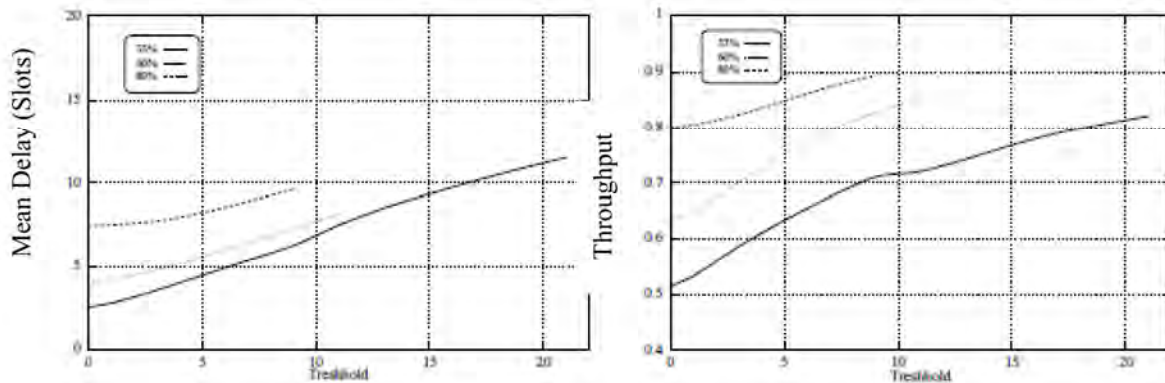
The mean delay shows the greatest increase with a higher load, since the polling competes most strongly with the use data for the channel and the use data uses less capacity. Moreover, as the **Acknowledge Threshold** increases, so does the probability that the transmission window will close, thereby causing further delays.

Thus, the following is true for the when traffic is unidirectional:

$$\text{Acknowledge Threshold} = \text{ARQ Timer Delay} - 1 \tag{8.2}$$

The throughput can be increased even further by using a greater **ARQ Timer Delay**, to the detriment of the mean delay, or by using a larger window, to the detriment of the overhead.

The same simulation is now observed with bidirectional traffic, i.e. the simulation uses the parameters from Tale 8.2, with the same load on the uplink and downlink.



**Figure 8.3:** Mean Delay and Throughput when Traffic is Bidirectional

The diagrams in Figure 8.3 reveal that the mean delay initially decreases when the *Threshold* increases. It does not begin to rise again, based on the total load, until a *Threshold* of 24. The throughput, on the other hand, increases continuously, first strongly, then slightly.

In contrast to unidirectional traffic, the *Threshold* now affects the polling as well and it is also delayed. This causes the negative influence of the polling to be delayed and weakens the effects considerably since many of the polling requests can be transmitted piggyback. At first glance, the reduction of the mean delay by collecting acknowledgments seems to be contradictory. However, for one thing, the acknowledgements alone rarely take channel capacity away from the ATM cells and for another, the *Ignore Timer* can be used to achieve a positive evaluation of the delayed acknowledgment (chapter 5.2.2). Thus, the following dimensioning rule would be advantageous for bidirectional traffic:

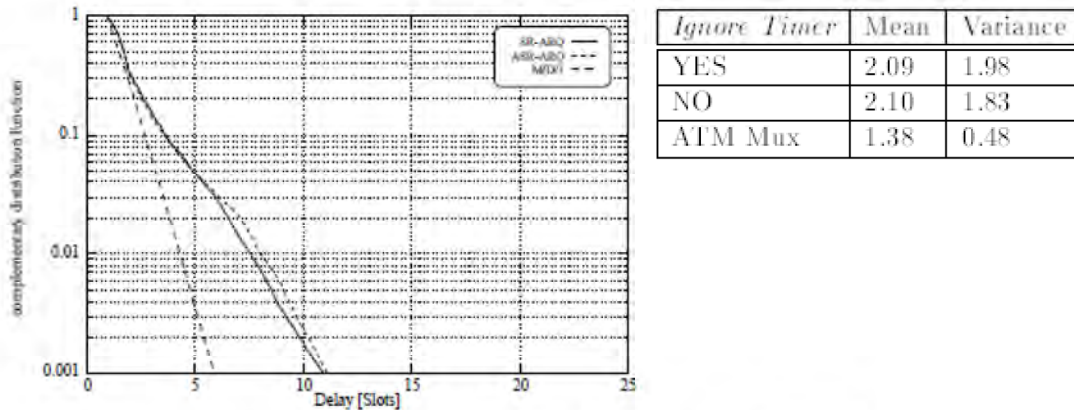
$$\text{Acknowledge Threshold} \approx 2 \cdot \text{ARQ Timer Delay} \tag{8.2}$$

Finally, I would like to point out the significance of the *Threshold*, based in this case on the throughput with a 33% channel load. Increasing the throughput from 52% to 80% creates a savings in transmission performance of  $1 - \frac{52\%}{80\%} = 35\%$ !

### 8.3 Examining the Causes of Cell Delays

#### 8.3.1 Influence of the Ignore Timer

To understand the influence of the *Ignore Timer* on the cell delay, the protocol was compared with and without the *Ignore Timer* in very simple scenarios using an idealized ATM Multiplexer. In these simple scenarios, an MS operates a VC with a supply of 35%, 70% and 90% on the uplink using a Poisson source. The maximum delay is  $200 T_{slot}$  and the *ARQ Timer Delay* is set to  $2 T_{slot}$  (minimum value).



**Figure 8.4:** Cell Delay at 35% Supply

Both variants behave in a very similar manner when the supply is only 35%. Whereas the *Ignore Timer* does create a smaller delay, the variance of the delay is smaller without it (table shown in Figure 8.4). The differing variance can be seen in the diagram in Figure 8.4 where the curves shift apart when the cell delay is greater than  $6 T_{slot}$ .

The deviation of the ARQ protocols from the ATM Multiplexer is caused by the protocol, i.e. the cause lies in the simulation of the ATM Multiplexer. The package error rate is simulated in the ATM Multiplexer by not servicing any requests at the level of the error rate. The delay in this case is  $1 T_{slot}$ . In contrast, the loss of a frame is not identified in the ARQ protocol until another frame is successfully received. Further, the receiver then has to inform the sender of the loss. The delay is thus at least  $3 T_{slot}$ . Since other frames were transmitted in the meantime, there is no jump in the displayed curves.

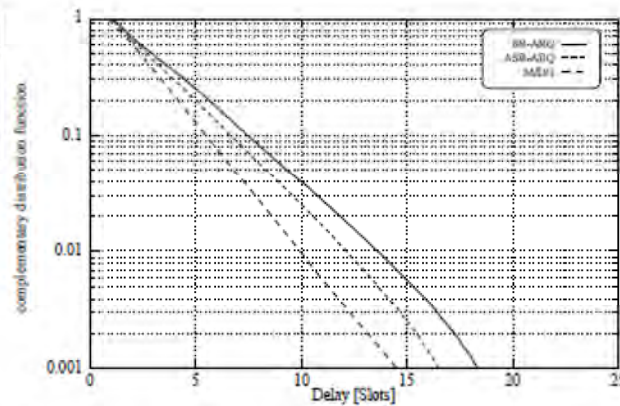
Significant differences can be seen in the protocol variants when the supply is 70%. Whereas the performance decreases slightly compared to the ATM Multiplexer when the *Ignore Timer* is not used, the behavior of the protocol improves when it is used. There are three effects that play a role in this:

1. By increasing the supply the ARQ protocols can continue transmitting frames while the frame loss is being dealt with more often. Thus, the loss of a frame does not affect the process as strongly as it would when the supply is only 35%. That is why there are no turning points in the curves in Figure 8.5.
2. The poorer behavior of the protocol when the *Ignore Timer* is not used is the result of the fact that it begins to stutter when a frame is lost. Stuttering means, that the receiver's repeat request

for

a

<i>Ignore Timer</i>	Mean	Variance
YES	3.47	6.18
NO	3.83	7.98
ATM Mux	2.59	3.80



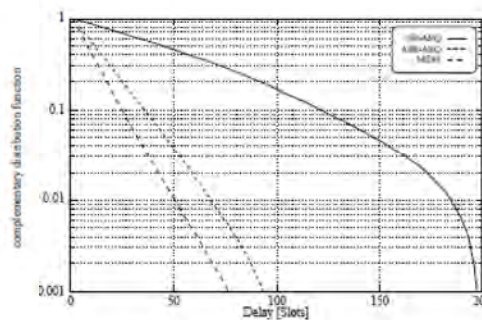
**Figure 8.5:** Cell Delay when Supply is 70%

frame using SREJ causes the sender to send the frame multiple times (in this case twice), even though it is unlikely that it will be lost again. That is why a higher *ARQ Timer Delay* should be selected even when the *Ignore Timer* is not used. The stuttering is reflected by the rate of multitransmissions. Whereas the rate is less than 0.2% when the *Ignore Timer* is used, the rate of 3.9% seen when the *Ignore Timer* is not used is within the package error rate range. Thus, stuttering is seen in 80% of the cases.

- Information on lost frames can be derived from positive acknowledgements with the *Ignore Timer* is used. This allows a faster reaction to the loss. This effect can be seen in the LLC overhead. Whereas the overhead is 13.4% when the *Ignore Timer* is not used, the overhead is 6.1% with the *Ignore Timer*, just slightly above the package error rate of 5%.

The table in Figure 8.5 shows the effect of the described behavior on the mean delay and the variance of the delay.

With a supply of 90%, the ARQ protocol has already almost reached the limits of its performance without the *Ignore Timer*. Minor losses can already be seen in the simulated scenario, i.e. individual cells exceeded the maximum delay and were discarded. Limiting the delay to no more than 200  $T_{slot}$  also explains the turn in the curve shown in Figure 8.6. The performance can only be increased without using the *Ignore Timer* by using a larger window.



**Figure 8.6:** Cell Delay when Supply is 90%

With the *Ignore Timer* the ARQ protocol shows only a slightly poorer performance than the ATM Multiplexer. The reasons for the different performances are shown in Table 8.3

When the *Ignore Timer* is used, the ARQ protocol has a multitransmission rate that is 10 to

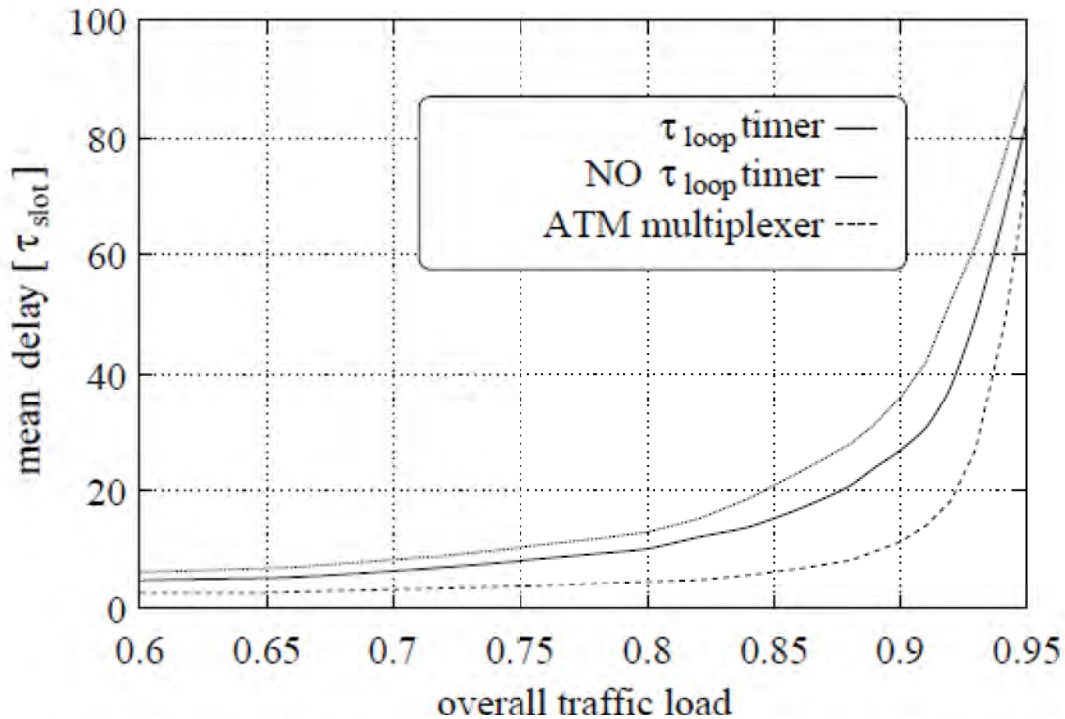


<i>Ignore Timer</i>	Mean Delay	Variance	Multitransmission	Overhead
YES	16.22	227.16	0,2%	5,4%
NO	55.66	1998.28	3,9%	8,9%
ATM Mux	11.03	112.79	-	-

**Table 8:** Mean Delay, Variance, Multitransmission Rate and Overheat when the Supply is 90% 20 times smaller and thus a much smaller protocol overhead. This overhead is composed of three parts:

1. Multitransmissions in the amount of the package error rate. These are absolutely necessary.
2. Multitransmissions due to the protocol behavior. These are superfluous and place additional load on the channel.
3. Transmission of supervisory frames without use data. These are necessary in order to poll the receiver when the transmission window is closed. They also place an additional load on the channel and should be avoided.

Finally, the performance of the *Ignore Timer* ( $\tau_{loop}$  Timer) is demonstrated in a realistic scenario.



**Figure 8.7:** Mean Delay Across the Entire Supply when the *Ignore Timer* Is Used

A scenario with 6 MS is now used, each of which operate a VC with symmetric load. The load is generated by Poisson sources. The diagram in Figure 8.7 shows the transmission delays across the supply with and without consideration of the *Ignore Timer*. The idealized ATM Multiplexer was used

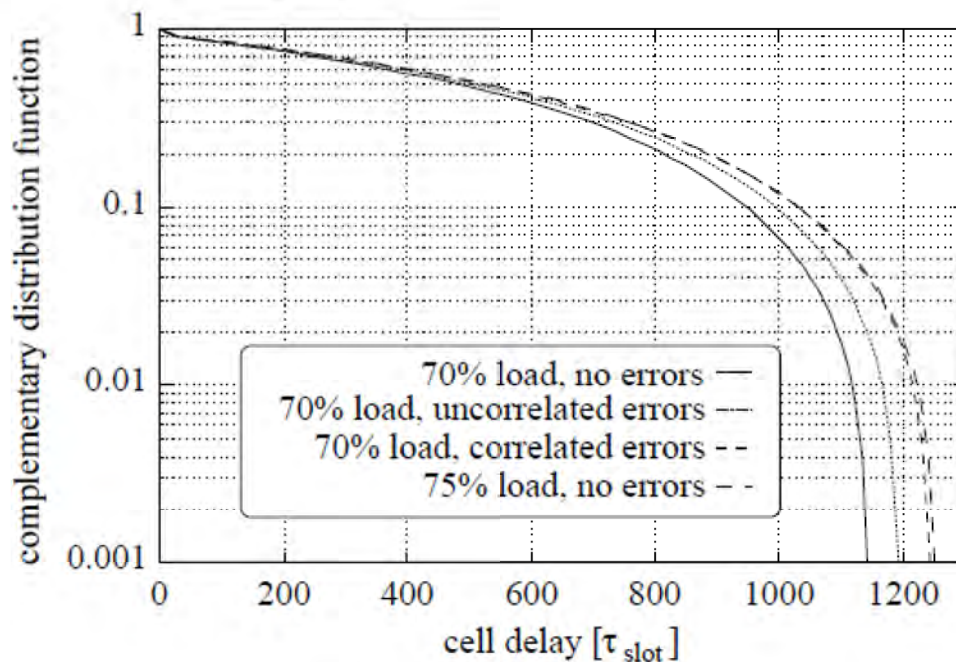
for the lower limit. The diagram demonstrates that the mean delay is reduced considerably when the *Ignore Timer* is used.

### 8.3.2 Influence of the Error Model

There are two causes for transmission delays when transmitted ATM cells:

1. Individual ATM cells are delayed significantly due to multitransmissions.
2. The multitransmission of individual missing cells creates an addition load, which causes all ATM cells to be delayed.

The multitransmission of ATM cells is shown preferable treatment in the used priority algorithm since the repeated transmission is closer to its deadline than a new transmission. That is why the second cause has a decisive influence on the cell delay. To demonstrate this behavior in a simulation, a scenario is examined that has five mobile stations, each with one VC, and with traffic on the uplink only. The load is generated by video sources with a mean data rate of 14% of the channel capacity (corresponds to 5Mbit/s in a 34Mbit/s channel). The total supply thus takes up 70% of the channel capacity. Three different error models are examined:



**Figure 8.8:** Cell Delay with Different Error Models

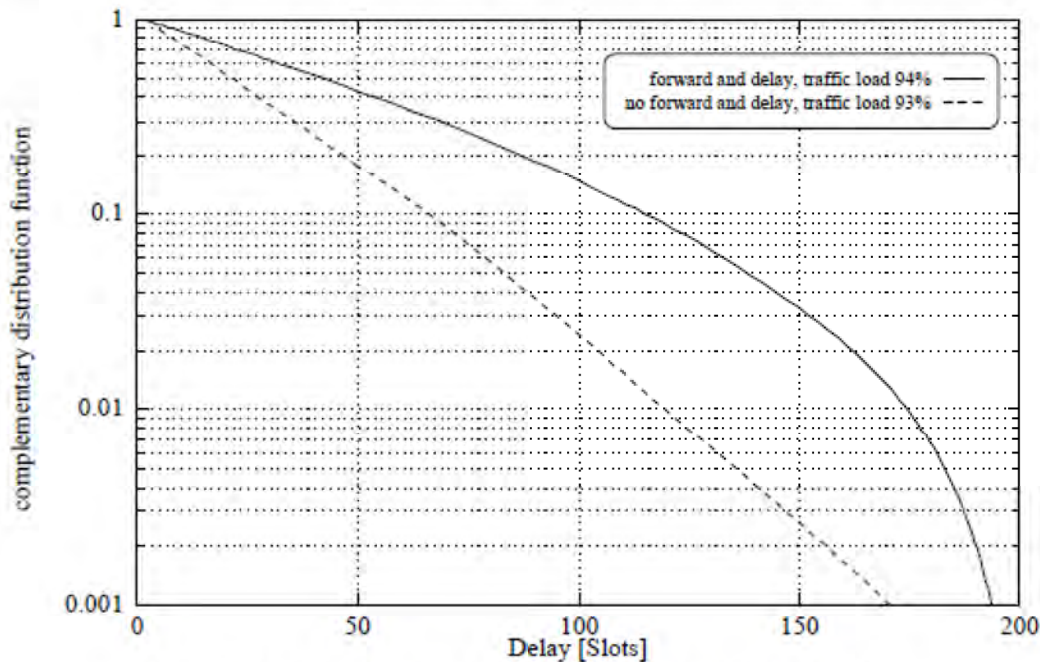
- No bit errors
- Uncorrelated bit errors with a rate of  $10^{-4} \equiv 5\%$  frame error rate
- Correlated bit error based on the Gilbert model (chapter 7.4) with a rate of  $10^{-3}$  in the **Error** state, corresponds to a mean frame error rate of 5%

Figure 8.8 shows that the delay increases by less than 5% when errors are uncorrelated and less than 10% when errors are correlated. If these curves are compared to a curve where the total load increases by 5% with an error-free model, it can be determined that the delay corresponds to the one where correlated errors occurred. This result was expected, since the additional load from the video sources is just as correlated as the additional load that is caused by the correlated errors.

#### 8.4 Examining the Effort Involved in Discarding Cells

Considerable effort is required to be able to discard cells that have exceeded their remaining life cycle in order to maintain the exact maximum cell delay for the ATM cells. These efforts are:

1. Use of a *Delay* timer to monitor the remaining life in the sender
2. Use of a *Forward* timer to monitor the remaining life in the receiver. Once this timer has run out, the corresponding cell is forwarded to the upper layer without waiting for missing cells.
3. Use of *Delay* PDUs to inform the receiver that cells were discarded.
4. Use of the *delay time* field (Figure 5.6) to transmit the remaining life.



**Figure 8.9:** Notification of the Receiver that Cells Are Late

The following demonstrates a simplified approach:

1. The remaining life cycle is checked before the cells are transferred to the transmission window. If there is absolutely no chance that the cell will be transmitted before the end of its life cycle, it is discarded.

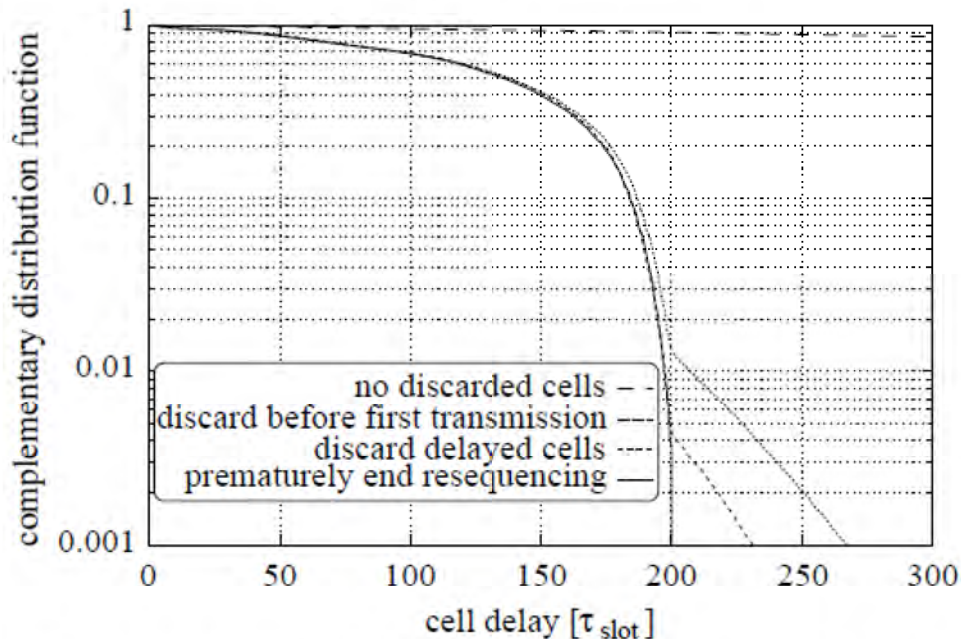
2. If the cell's life cycle expires when it is in the transmission window, it is not discarded and is transmitted anyway. Since the transmission window is limited (4-8 slots), tight limits are also set for exceeding the delay.

The first thing that is examined is whether it makes sense to code the remaining life cycle in 6-8 bits, or whether the additional overhead is too large. To do so, a scenario with one MS that has one VC with use data on the uplink one is examined. The supply takes up 94% of the channel capacity. A coding in 6 bits corresponds to an overhead of 1%. The additional overhead is then accounted for in a manner reduces the supply by 1% for a protocol where the remaining life is not transmitted.

Figure 8.9 clearly shows that the additional overhead leads to considerably poorer results. At least in this scenario, which is not very realistic, it does not appear reasonable to transmit the remaining life cycle.

The methods presented in chapter 5.2.3 for handling time-critical ATM cells will now be examined in a more realistic scenario. These methods are:

1. No consideration of the maximum delay
2. Discarding before the first transmission
3. Discarding of all delayed cells and use of the Delay PDU
4. Premature cancellation of the re-sequencing in the receiver



**Figure 8.10:** Cell Delay Using Different Strategies for Discarding Cells

To do so, a scenario with six mobile stations, each with one VC, is examined. Each VC operates one Poisson source for the uplink and downlink. The total supply takes up 95% of the channel capacity, which creates an overload when there is mean frame error rate of 5%. This simulation can be interpreted as a short-run overload situation in a system that is otherwise operated with a lower total supply.

procedures	label in Figure 8.10	% discarded cells	% lost cell
-	no discarded cells	0	90.1
1	discard before first transmission	1.85	3.07
1-2	discard delayed cells	2.31	2.72
1-3	prematurely end resequencing	1.76	1.76

**Table 8.4:** Discarded and Loss Cells as a Percentage (See Figure 8.10)

The curves shown in Figure 8.10 are part of the strategies shown in Table 8.4. The *discard ratio* and CLR are also shown in this table.

If no ATM cells are discarded, 90% of the cells exceed the maximum delay of  $200 T_{slot}$ . The more cells that are discarded, the smaller the cell delay of the successfully transmitted cells is. If the ATM cells exceed the maximum delay, they are counted as lost. Table 8.4 shows that the cell loss rate decreases as the complexity of the protocol increases.

When methods 1-3 are used, the receiver has exact knowledge of the need for acknowledgements, which is why fewer cells are discarded in this case. As soon as the resequencing is cancelled, the receiver sends an acknowledgement. This frequently makes it unnecessary to transmit Delay PDUs, in contrast to situations when only methods 1-2 are used.

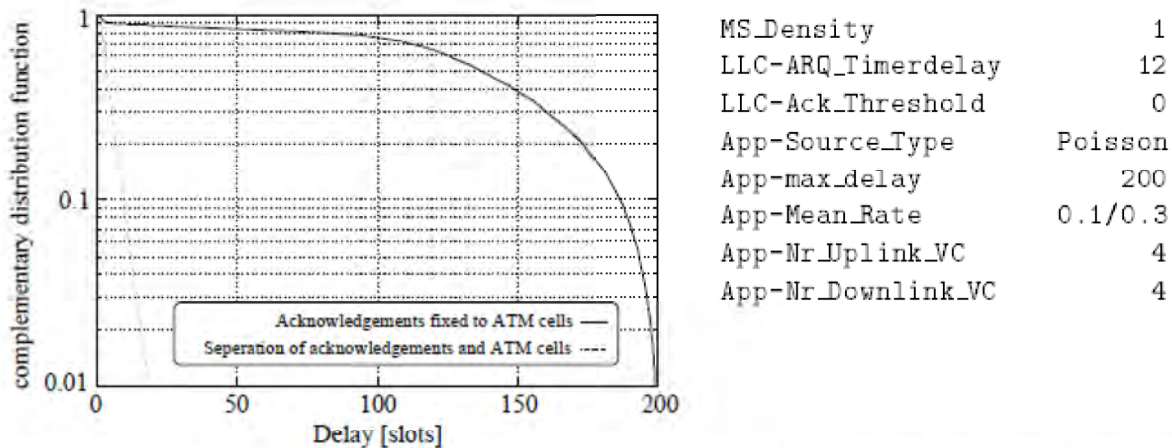
The gradient of the branches of the curves when the maximum delay is exceeded depends on which priority the delays cells or Delay PDUs are given. In this case, they are straight lines since no cells are discarded in this section. When methods 1-2 are used, this branch of the curve contains the cells that are in the receiver, but are waiting for late cells. The gradient of the curve depends exclusively on the priority of the Delay PDUs, which were set very high in this scenario. The high priority results in an increased *discard ratio*, since the Delay PDUs may take capacity away from the use data. When using method 1, the gradient of the branch depends exclusively on the priority of the late cells, which is very high when the Scheduler is used, since the priority is calculated based on the deadline alone.

It would be advantageous to use all three methods if the remaining life cycle of the cells could be coded into just a few, or even a single bit. Corresponding tests would have to be performed to do so.

### 8.5 Separating Acknowledgements and Use Data

This simulation shows what sort of influence separating acknowledgements and use data has. To demonstrate this, a scenario with one mobile station that has four virtual channels is examined. Each of these channels is bidirectional and is fed on both sides Poisson sources. Two of these channels have a mean data rate of 30% on the uplink and 10% on the downlink. The traffic flow in the other two channels is the exact opposite. In one case, use data can only transport acknowledgments from the same channel, while in the other case, from any channel.

Figure 8.11 clearly shows that when acknowledgements are bound to the same channel as the use data the protocol is no longer able to meet the deadline.



**Figure 8.11:** Cell Delay when Acknowledgements and Use Data Are Connected or Separated

There are considerable losses (10%) even though only 80% of the capacity of the uplink and the downlink is being used.

The most urgent acknowledgements are always sent when the acknowledgement channel can be freely selected when transmitting ATM cells. This leads to a smooth protocol flow with minimal delays.

Multiple parallel VCs can only be operated in a mobile station with a high channel load, without separating acknowledgments from use data, when there is a bidirectional, symmetrical supply. However, this condition restricts the service offer significantly.

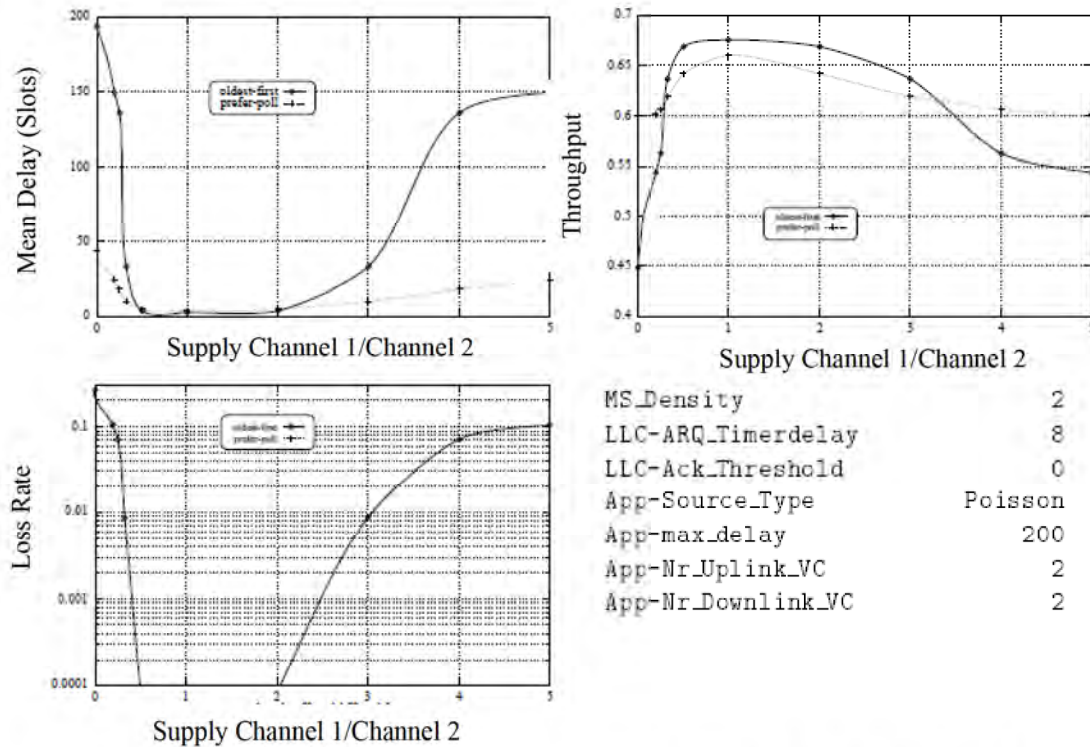
## 8.6 Asymmetric Traffic Flow

The influence of the priority algorithms for acknowledgments (**ARQ Receive Priority**) on the mean delay, throughput and loss rate is now examined. A scenario with two mobile stations is used to do so. Each mobile station operates two virtual channels. One channel has a use data flow on the uplink and the other on the downlink. Poisson sources are used and there is a total data flow of 60%. The distribution of the total supply is varied on the uplink and downlink in a range between 0-5. In doing so, the stations carry loads in opposing directions, i.e. the main load of station 1 is on the uplink when the main traffic in station 2 is on the downlink.

This scenario produces the following problems for the acknowledgement process:

1. The supply on the reverse channel of the heavily-loaded channel is not sufficient for transmitting acknowledgements by piggyback.
2. Pure acknowledgements (without use data) compete with the use data of the other heavily-loaded channel for use of the medium.
3. If the heavily-loaded channel is not acknowledged in time, its transmission window closes and it begins to poll; which places an additional burden on the channel.

The two priority algorithms examined use different methods for increasing the priority of the acknowledgements:



**Figure 8.12:** Mean Delay, Throughput and Loss Rate when Traffic Flow is Asymmetric

**oldest\_first** This algorithm increases the priority mainly based on the wait time of the acknowledgements.

**prefer\_poll** Answers to polls and SREJ acknowledgements are automatically assigned a high priority. The priority of the RR acknowledgements increases quickly as the number of frames requiring acknowledgement increases.

On average, the acknowledgements in **prefer\_poll** have a much higher priority than those in **oldest\_first** and are thus able to prevail over use data more often.

Figure 8.12 shows that **oldest\_first** behaves more favorably than **prefer\_poll** when there is lower asymmetry in the channels, both in regard to delays, as well as the throughput. In case of greater asymmetry, **prefer\_poll** is merely able to guarantee the minimum throughput of 60% in order to prevent losses. The **oldest\_first** algorithm causes the protocol to collapse under these settings, resulting in considerable losses.

The causes for the collapse can be explained based on the results of the absolute asymmetric supply (in this case the second channel does not have use data). When the load is 60% and the error rate is 5%, 35% of the capacity is still available to transmit pure acknowledgements, i.e. when the capacity is used in full, an average of every other frame could be acknowledged. Using a window size of 4 should thus make the process more stable.

Table 8.5 shows that the use data competes with a large number of acknowledgements. 4.7% of the capacity is used by the polling process alone. The low mean wait time of the pure acknowledgements shows that they are transmitted when use data cannot be transmitted. On the other

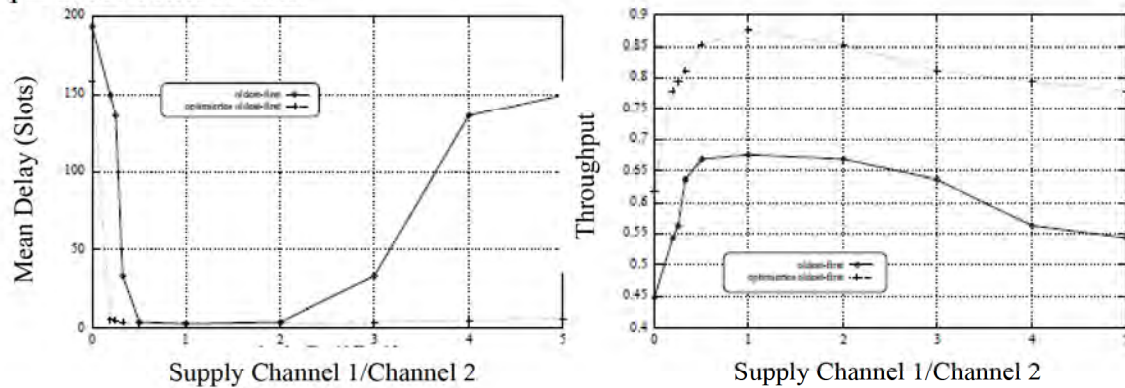
	Station 1		Station 2	
	Uplink	Downlink	Uplink	Downlink
Supply	60%	-	-	60%
Capacity used	56.6%	36.5%	36.5%	56.5%
Use data	51.8%	-	-	51.8%
Pure acknowledgements	4.7%	36.5%	36.5%	4.7%
Mean wait time for pure acknowledgements	1.05	1.57	1.57	1.05

**Table 8.5: oldest\_first Measurements**

hand, the mean delay of the ATM cells is very high in comparison, which means this situation always occurs when the transmission window is closed. The collapse of the protocol can thus be explained by the fact that the channels block one another when transmitting acknowledgements when the window is closed.

As long as an error does not occur, the transmission windows both close at the same time and enable the transmission of acknowledgements. However, if an error does occur, one window is closed later and that channel does not receive an acknowledgement until the other window closes or until the acknowledgement has waited long enough. In this case, however, the transmission window rarely closes, since the reverse channel is now available to the acknowledgements.

It will now be examined whether bundling the acknowledgements based on equation 8.2 (chapter 8.2.2), while optimizing the **ARQ Timer Delay** based on equation 8.11 (chapter 8.2.1) produces better results.



**Figure 8.13: Mean Delay and Throughput of the Optimized oldest\_first**

Figure 8.13 shows that enhancing the parameters results in much more favorable behavior in the protocol. The mean delay remains low, even when the asymmetry increases, and the throughput increases considerably throughout the entire area, averaging an absolute of 20%.

Despite a throughput of more than 60%, there are considerable losses with the asymmetry (13%) and the protocol collapses. The reasons for the collapse can be seen in Table 8.6. 11% of the capacity is consumed by polling, a clear indication that acknowledgements are not being transmitted in time and that an answer is not sent until the polling has been repeated numerous times.

It will now be examined whether a more hybrid approach is successful. To do so, **oldest\_first** is expanded to the extent that answers to polls are automatically assigned a high priority. The new



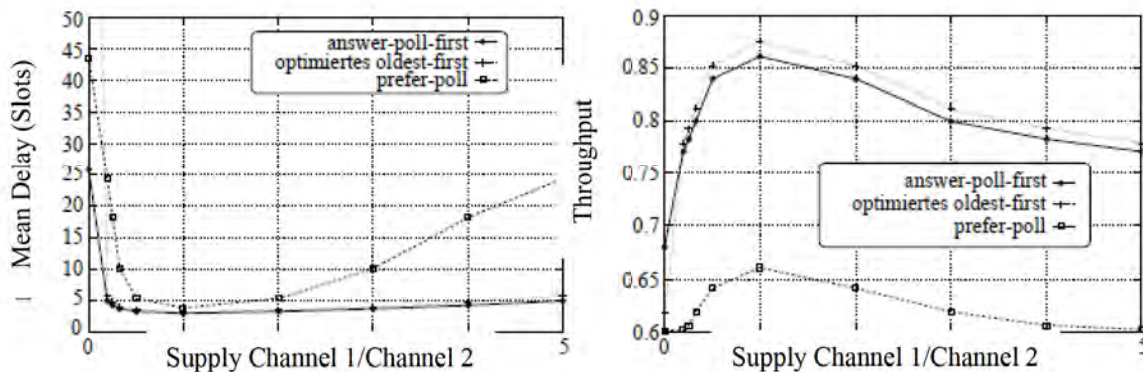
	Station 1		Station 2	
	Uplink	Downlink	Uplink	Downlink
Supply	60%	-	-	60%
Capacity used	63.6%	17.4%	17.3%	63.6%
Use data	52.6%	-	-	52.7%
Pure acknowledgements	11.0%	17.4%	17.3%	11.0%
Mean wait time for pure acknowledgements	1.05	4.23	4.24	1.05

**Table 8.6:** Measurements of the Optimized **oldest\_first**

algorithm will be called **answer\_poll\_first** and should have the following properties:

- Delay and throughput of the optimized **oldest\_first**
- Stability of **prefer\_poll**

Figure 8.14 shows that the requirements for **answer\_poll\_first** can be easily met. This algorithm generates the smallest mean cell delay in all of the measurements and the throughput is only slightly worse than the optimized **oldest\_first**. The lower throughput is caused by the transmission of pure acknowledgements in answer to the polls. That is also why the mean cell delay is lowest.



**Figure 8.14:** Mean Delay and Throughput of **answer\_poll\_first**

A final optimization of the acknowledgement algorithm is only possible in connection with a real MAC protocol. The special features of the respective MAC protocol do have to be taken into consideration at that point. However, it should be noted that in idealized conditions, the best results are achieved when normal acknowledgements are bundled and poll answers and Delay PDUs a given high priority (see chapter 8.4). This should be taken into consideration in future developments.

## Summary and Prospects

In this paper, the ASR ARQ protocol was enhanced and the performance thereof was evaluated. Following a brief introduction into MBS and ATM, the need for error correction processes in the radio interface was explored. The requirements for an ARQ protocol and the resulting structure of the LLC layer were then presented. Further, the properties of the MAC layer that could be used advantageously by the ARQ protocol were also explored and the main properties were the dynamic capacity assignment, the deterministic transmission delay and the fact that the ARQ frames are not stored temporarily in the MAC layer.

Following an overview of the conventional ARQ protocols, the expansions compared to standardized protocols were presented. The most significant are the operation of parallel ARQ instances, the separation of use data and acknowledgements, the use of the *Ignore Timer* and the special treatment of time-critical ATM cells.

The ASR ARQ protocol was designed as a finite state machine, was implemented in C++ and the resulting class hierarchy was demonstrated. Furthermore, modified application and MAC layers were implemented to evaluate the performance and the services provided by their layers were provided in an idealized manner in order to eliminate external influences when evaluating the performance.

The ASR ARQ protocol was evaluated through simulation in different scenarios using the ATM performance parameters. The use of the *Ignore Timer* proved to be useful in the simulations. Further investigation should show the extent to which the assumed constant transmission delay exists in a real system.

The operation of parallel ARQ instances for virtual channels with different characteristics and the separation of acknowledgements and use data proved to be very advantageous. Despite the increased overhead, considerable improvement as seen in the cell delay.

Different methods were developed to discard time-critical ATM cells once they have exceeded their maximum delay. The simulations showed that discarding cells is very useful in solving short-term overload situations. Parameters can be set for the complexity of the ASR ARQ protocol, in regard to the discard method, enabling different approaches to be used depending on the available capacity. The question of coding remaining life cycle is, however, still just as open as the question of the stability of the extended protocol. An analytical examination of the developed protocol would probably be useful.

Finally, the need for intelligent protocol algorithms for sending acknowledgements was demonstrated. The special features of the used MAC layer should be considered in the continued development of these algorithms.

The ASR ARQ protocol must be adjusted to the existing MAC protocols and the MAC protocols still in development. A renewed evaluation of the performance is necessary to verify the assumptions made herein. The use of shorter signaling slots in the DSA++ protocol gives the ASR ARQ protocol the possibility to send acknowledgements, even without ARQ frames. Thus, the protocol should be modified.

The ASR ARQ protocol is perfectly suited for handling the services of the VBR and ABR classes. An alternative, simple protocol that discards a cell once two attempts to transmit it have been made, could be used for the CBR service class. What remains to be examined is what sort of structure such a protocol should have to be advantageous and whether the protocol is already covered by the ASR ARQ protocol when corresponding parameter settings are defined. If not, it would need to be examined whether the implementation of two independent protocols is justified or whether the greater effort involved in the ASR ARQ protocol would be required for the CBR services to maintain a uniform protocol structure.

---

**List of Abbreviations**

AAL	ATM Adaptation Layer
ABR	Available Bit Rate
ADCCP	Advanced Data Communication Control Procedure
ADM	Asynchronous Disconnected Mode
ARQ	Automatic Repeat ReQuest
ASR	Adaptive Selective Repeat
ATM	Asynchronous Transfer Mode
B-ISDN	Breitband-ISDN
BSC	Base Station Controller
BST	Base Station Transceiver
BTS	Base Transceiver Station
CAC	Connection Admission Control
CBR	Constant Bit Rate
CDV	Cell Delay Variation
CEP	Connection End Point
CER	Cell Error Ratio
CLP	Cell Loss Priority
CLR	Cell Loss Ratio
CMR	Cell Misinsertion Rate
CNCL	Communication Networks Class Library
CPCS	Common Part Convergence Sublayer
CS	Convergence Sublayer
CTD	Cell Transfer Delay
DCS 1800	Digital Cellular System 1800
DECT	Digital European Cordless Telecommunications
DM	Disconnect Mode
FCFS	First Come First Serve
FDM	Frequency Division Multiplexing
FDMA	Frequency Division Multiple Access
FEC	Forward Error Correction
GCCH	Global Control CHannel
GFC	Generic Flow Control
GIST	Graphical Interactive Simulation result Tool
GSM	Global System for Mobile Communication
HDLC	High level Data Link Control
HEC	Header Error Control
IBCN	Integrated Broadband Communication Network
ISDN	Integrated Services Digital Network
LAP-B	Link Access Procedure-Balanced
LLC	Logical Link Control
LLI	Logical Link Identifier
LRE	Limited Relative Error

MAC	Medium Access Control
MBS	Mobile Broadband System
MBT	Mobile Broadband Termination
MCTD	Mean Cell Transfer Delay
MS	Mobile Station
NNI	Network Node Interface
PDU	Protocol Data Unit
PT	Payload Type
QAM	Quadrature Amplitude Modulation
QPSK	Quaternary Phase Shift Keying
QoS	Quality of Service
RACE	Research and technology development in Advanced Communications technologies in Europe
REJ	REJect
RN	Request Number
RR	Receive Ready
SABME	Set Asynchronous Balanced Mode Extended
SAP	Service Access Point
SAR	Segmentation And Reassembly
SDLC	Synchronous Data Link Control
SDU	Service Data Unit
SECBR	Severely Errored Cell Block Ratio
SN	Sequence Number
SP	Service Primitive
SREJ	Selective REJect
SR	Selective Repeat
SSCOP	Service Specific Connection Oriented Protocol
SSCS	Service Specific Convergence Sublayer
STDM	Synchronous Time Division Multiplexing
TCH	Traffic CHannel
TDMA	Time Division Multiple Access
UA	Unnumbered Acknowledgment
UMTS	Universal Mobile Telecommunication System
UNI	User Network Interface
VBR	Variable Bit Rate
VCI	Virtual Channel Identifier
VC	Virtual Channel
VPI	Virtual Path Identifier

---

 TABLE OF FIGURES
 

---

2.1	Static Multiplexers in an ATM Connection.....	7
2.2	ATM Cell Header.....	7
2.3	Virtual Path Switching and Virtual Channel Switching.....	8
2.4	ATM Reference Model.....	9
2.5	Classification of the Services in the ATM Adaptation Layer.....	10
3.1	MBS and Other Data Networks.....	11
3.2	Applications and Services of the MBS.....	12
3.3	Service-Specific ATM Transmission.....	13
3.4	Transparent, Mobile ATM Access.....	13
3.5	Layout of a Slot.....	15
3.6	Cell Structure of the MBS System.....	16
4.1	Distribution of the Different VCs to ARQ Instances.....	19
4.2	Layout of the LLC Layer in the Mobile Station .....	22
4.3	Layout of the LLC Layer in the <i>Base Station Controller</i> .....	23
5.1	Example of Transmitting RR and SREJ.....	31
5.2	Avoiding Unnecessary Repetitions by Using the <i>Ignore Timer</i> .....	32
5.3	Fast Reaction to Losses by Using the <i>Ignore Timer</i> .....	33
5.4	Deadlocks Caused by Prioritizing Delay PDUs.....	35
5.5	Deadlocks Caused by Prioritizing Acknowledgements.....	36
5.6	Frame Structure of the ASR ARQ.....	37
5.7	Generating Piggyback PDUs.....	37
5.8	Events for Triggering Acknowledgements.....	38
6.1	Structure of the Upper LLC Layer.....	40
6.2	<b>ConnectionHandler</b> States.....	42
6.3	Events when an Acknowledgement is Received.....	43
6.4	<b>Send_Data_Object</b> States.....	45
6.5	Events when a Frame Is Received.....	47

6.7	<b>Receive_Data_Object</b> States.....	48
7.1	Simulation Modules.....	50
7.2	Application Layer in the Mobile Station.....	52
7.3	Application Layer in the Base Station.....	53
7.4	Model in the $H_2$ Distribution.....	55
7.5	Autoregressive Process.....	56
7.6	Gilbert Model.....	57
7.7	Screen Display in GIST.....	60
7.8	Elements in GIST.....	61
8.1	Mean Delay, Variance and Throughput with Variation of ARQ Timer Delay.....	65
8.2	Mean Delay and Throughput when Traffic Is Unidirectional.....	66
8.3	Mean Delay and Throughput when Traffic is Bidirectional.....	67
8.4	Cell Delay at 35% Supply.....	68
8.5	Cell Delay at 70% Supply.....	69
8.6	Cell Delay at 90% Supply.....	69
8.7	Mean Delay Across the Entire Supply when the <i>Ignore Timer</i> Is Used.....	70
8.8	Cell Delay with Different Error Models.....	71
8.9	Notification of the Receiver that Cells Are Late.....	72
8.10	Cell Delay Using Different Strategies for Discarding Cells.....	73
8.11	Cell Delay when Acknowledgements and Use Data Are Connected or Separate.....	75
8.12	Mean Delay, Throughput and Loss Rate when Traffic Flow is Asymmetric.....	76
8.13	Mean Delay and Throughput of the Optimized <b>oldest_first</b> .....	77
8.14	Mean Delay and Throughput of <b>answer_poll_first</b> .....	78

---

 TABLE OF TABLES
 

---

3.1	Parameters of the MBS System.....	14
6.1	State Transitions of the <b>Send_Data_Object</b> .....	46
6.2	State Transitions of the <b>Receive_Data_Object</b> .....	46
7.1	Protocol Variants of the Different Layers in <b>.sim_defaults</b> .....	51
7.2	Service Primitive of the Application Layer.....	53
7.3	Mean Value and Variance by Service Class.....	55
7.4	Mean Value of the Number of Services per Hour.....	56
7.5	Parameters in the <b>.sim_defaults</b> Configuration File.....	58
7.6	Parameters in the <b>.uplink_vc_llc</b> Configuration File.....	59
8.1	Overview of the ASR ARQ Parameters.....	64
8.2	Simulation Parameters.....	66
8.3	Mean Delay, Variance, Multitransmission Rate and Overheat when the Supply is 90%..	70
8.4	Discarded and Loss Cells as a Percentage (See Figure 8.10).....	74
8.5	Mean Delay and Throughput of the Optimized <b>oldest_first</b> .....	77
8.6	Measurements of the Optimized <b>oldest_first</b> .....	78



## BIBLIOGRAPHY

- [1] C. Apostolas, R. Tafazolli, B. G. Evans. *Wireless ATM LAN*. In *PIMBC'95*, pp. 773–777, Toronto, Canada, September 1995.
- [2] D. Bertsekas, R. Gallager. *Data Networks*. Prentice-Hall, Englewood Cliffs, NJ, 1987.
- [3] J. Brázio, C. Belo, S. Sveat, B. Langen, D. Plassmann, P. Roman, A. Cimmino. *MBS Scenarios*. In *RACE Mobile Workshop*, pp. 194–197, Metz, F, June 1993.
- [4] J. Brázio, J. Sobrinho, Correia A. *Multiple access methods for packet video*. R2067 MBS/WP2.2.2/IST032.2, IST, February 1994.
- [5] O. Casals, J. Garcia, C. Blondia. *A Medium Access Control Protocol for an ATM Access Network*. In *Proc. of 5th Int. Conf. on Data Comm. Syst. and their Performance*, Raleigh, North Carolina (USA), October 1993.
- [6] B. Desealers. *Structure and Protocols of the Security Layer for a Mobile Extension of ATM Networks Under Consideration of Multilink Transmission and Handovers*. Diploma Thesis, Communication Networks, RWTH Aachen. 95.
- [7] R. Dinis, A. Gusmão. *Adaptive Serial OQAM-Type Receivers for Mobile Broadband Communications*. In *IEEE 45th Vehicular Technology Conference*, pp. 200–205, July 1995.
- [8] L. Fernandes. *Developing a System Concept and Technologies for Mobile Broadband Communications*. IEEE Personal Communications, Vol. 2, No. 1, pp. 54–59, February 1995.
- [9] Galagher. *Networks*. Addison Wesley, 1993.
- [10] C. Garburg. *Development and Performance Evaluation of an ARQ Protocol for a Mobile Extension of ATM Networks Under Consideration of Multichannel Transmission and Macro-Diversity*. Diploma Thesis, Communication Networks, RWTH Aachen, March 95.
- [11] Schreiber F. Görg C. *Stochastic Simulation Technique*. Lecture at the RWTH Aachen, Aachen, 1995.
- [12] R. Händel, M.-N. Huber, S. Schröder. *ATM Networks: Concepts, Protocols, Applications*. Addison-Wesley, 1994.
- [13] T. Henderson. *Design principles and performance analysis of SSCOP: a new ATM Adaptation protocol*. Computer Communication Review, Vol. 25, No. 2, pp. 47–59, April 1995.
- [14] J. Hyman, A. Lazar, G. Pacifici. *Real-Time Scheduling with Quality of Service Constraints*. IEEE Journal on Selected Areas in Communications, Vol. 9, No. 7, pp. 1052–1063, September 1991.
- [15] J. Immonen, J.-M. Romann, D. Plassmann. *Requirements and Protocol Architecture for MBS access to ATM network*. In *RACE Mobile Telecommunications Summit*, pp. 324–328, Cascais, P, November 1995.
- [16] ISO. *Standard ISO 6256: HDLC – high level data link control*.

## Bibliography

- [17] ISO. *ISO 8802-2, Information processing systems-Local area networks-Part2:Logical link control*. International standard, ISO, 1990.
- [18] ITU-T. *Recommendation I.356: B-ISDN ATM Layer Cell Transfer Performance*, 1993.
- [19] ITU-T. *Recommendation I.363: B-ISDN ATM Adaptation Layer Specification*, 1993.
- [20] ITU-T. *Recommendation Q.2110: B-ISDN ATM Adaptation Layer - Service Specific Connection Oriented Protocol (SSCOP)*, 1993.
- [21] H. Kerner. *Computer Networks According to OSI*. Addison-Wesley, Munich, 1993.
- [22] Leonard Kleinrock. *Queuing Systems — Computer Applications*, Vol. 2. Wiley-Interscience, New York, NY, 1976.
- [23] H. Kopka. *LATEX: An Introduction*. Addison-Wesley, Bonn. 4<sup>th</sup> edition, 1993.
- [24] J.-Y. Le Boudec. *The Asynchronous Transfer Mode: a tutorial*. In *Computer Networks and ISDN Systems 24*, pp. 279 – 309, North-Holland, 1992.
- [25] H. Meineke. *Competitors Waiting for the Fall of the Telekom Monopoly. VDE-dialog, Vol. 1, No. 1, pp. 4-5, 1995*
- [26] S. Nogami. *A Study of Quality Control at the Cell Level in the ATM Network – Simplicity of Control Mechanisms vs. Efficient Utilization of Resources*. In *Proc. of the 13th Int. Teletraffic Congress*, pp. 987–992, Copenhagen, DK, June 1991.
- [27] D. Petras. *Functionality of the ASR-ARQ Protocol for MBS*. In *RACE Mobile Telecommunications Summit*, pp. 225–229, Cascais, P, November 1995. available at <http://www.comnets.rwth-aachen.de/~petras>.
- [28] D. Petras. *Medium Access Control Protocol for transparent ATM access in MBS*. In *RACE Mobile Telecommunications Summit*, pp. 218–224, Cascais, P, November 1995. available at <http://www.comnets.rwth-aachen.de/~petras>.
- [29] D. Petras. *Medium Access Control Protocol for wireless, transparent ATM access*. In *IEEE Wireless Communication Systems Symposium*, pp. 79–84, Long Island, NY, November 1995. available at <http://www.comnets.rwth-aachen.de/~petras>.
- [30] D. Petras, A. Hettich. *Performance evaluation of the ASR-ARQ Protocol for wireless ATM*. In *IEEE Wireless Communication Systems Symposium*, pp. 71–77, Long Island, NY, November 1995. available at <http://www.comnets.rwth-aachen.de/~petras>.
- [31] D. Petras, A. Hettich, B. Walke. *Design Principles and Performance Evaluation of an LLC Protocol for an ATM Air Interface*. In *IEEE Journal on Selected Areas in Communications*, Long Island, NY, January 1996.
- [32] S. Prata. *C++: Introduction to Object-Based Programming*. te-wi Verlag, Munich, 3<sup>rd</sup> edition, 1992
- [33] M. Prügler. *MBS Air Interface Principles*. In *RACE Mobile Telecommunications Summit*, Cascais, P, November 1995.
- [34] F. Reif. *C++ Implementing a Graphical Protocol Debugger for an ATM Radio Interface*. Student Research Paper, Communication Networks, RWTH Aachen, January 96.
- [35] L. Schmickler. *Ways to Use the Optimal Strategy SRPT in Local Computer Networks*. PhD Thesis, Communication Networks, RWTH Aachen, 91.
- [36] G. Siegmund. *ATM – The Technology of the Broadband ISDN*. V. Decker, Heidelberg, 1993.

- [37] P.F.M. Smulder. *Broadband Wireless LANs: A Feasibility Study*. PhD thesis, Eindhoven University of Technology, 1995.
- [38] B. Walke. *Improved bounds and an approximation for a dynamic priority queue*. In *3rd Int. Comp. Sympos. Modell & Perform. Evaluation of Computer Systems, Bonn, G, Oct. 1977*. North-Holland Publishing Company, 1977.
- [39] B. Walke. *Waiting-Time Distributions for deadline-oriented Serving*. In M. Arato, A. Butrimenko, E. Gelenbe, editors, *Performance of Computer Systems*, pp. 241 – 260. North-Holland Publishing Company, 1979.
- [40] B. Walke. *Communication Networks and Traffic Theory I*. Lecture at the RWTH Aachen, Aachen, 1993.
- [41] B. Walke. *Communication Networks and Traffic Theory II*. Lecture at the RWTH Aachen, Aachen, 1993.
- [42] B. Walke. *Mobile Networks and Their Protocols*. Lecture at the RWTH Aachen, Aachen, 1993.