MASTER

Chipcards in intelligent network services

van Helvoort, L.D.C.

Award date:
1998

Link to publication
Chipcards in Intelligent Network Services

L.D.C. van Helvoort

Coach: ir. J. van der Meer (Ericsson Rijen)
Supervisor: Prof.ir. J. de Stigter
Examiner: ir. M.J.M. van Weert
Period: November 1997
Preface

This report describes my graduation project for the Electrical Engineering studies at the Eindhoven University of Technology. The project was done at the Intelligent Network Application Laboratory (INAL), which is part of the R&D division of Ericsson Telecommunication in Rijen. Looking back, I can conclude that working for the INAL has been a great experience. It was nice to be part of such a big multinational as Ericsson is and to be working within the rapidly evolving telecommunications business. Besides expanding my knowledge on telecommunications and Intelligent Networks in particular, I was offered the possibility to explore the relatively new business area of chipcard technology. I value everything I learned and I am convinced that this experience forms a good starting-point for the rest of my career as an engineer.

However, all this would never have been possible without prof. ir. J. de Stigter, who believed in me and took the responsibility for my graduation project. His interest in my work was stimulating and his feedback on my work very useful. I would like to thank him for all that. I also want to thank J. van der Meer, who was my coach within Ericsson. He gave me the opportunity to do my graduation project at the INAL, and he was always very supportive and enthusiastic. I learned a great deal from him. I am very grateful to all my colleagues, for they were always very willing to help me and for making me feel a real INAL member. A special thanks to E. van der Velden, who put a lot of time and effort in helping me. Finally, I would like to thank my parents for their encouragements and for always supporting me.

November 1997
Leon van Helvoort
Summary

The Intelligent Network (IN) concept is more and more implemented in telecommunication networks all over the world. The standardization process for the IN has not finished yet. The IN concept is continuously enriched with new services and solutions. This continuing standardization ensures adaptation to recent requirements of the market and utilization of the newest technologies.

A new technology which might be interesting for IN is the chipcard technology. Chipcards tend to become very popular in all kinds of applications. The capability to store information on such a portable medium in a secure way, combined with the on-the-card processing power, especially useful for cryptographic calculations, provides a device which offers many possibilities.

The use of chipcards in IN services can be twofold. Firstly, all kind of user and service information can be stored on a chipcard. User information can be used for identification purposes. Service information on the card enables management and portability of service profiles, for instance to other operator’s networks. By means of a user interface the content of chipcards can be read, edited and stored. Secondly, thanks to their cryptographic capabilities, chipcards offer interesting security solutions. Besides secure transmission of information also reliable authentication of all involved parties is possible.

Many of the IN services that have been standardized up till now, can be made more secure, more user friendly and portable by means of a chipcard. To get some experience with chipcards in general and with the use of chipcards in IN services, a prototyping environment has been developed. The environment consists of a PC with a user interface, connected to a chipcard terminal and a modem, and IN functionality implemented on a Unix platform. To enable communication between the user interface and the IN, several new Service Independent Building Blocks (SIBs) and service scripts have been developed and new communication messages have been introduced.

In this environment the following IN/chipcard (IN/CC) services have been implemented:

- Abbreviated Dialing service
- Account Card Calling service
- Universal Personal Telecommunications service

The first service makes use of the storage capability, the latter two make use of the security capabilities of chipcards.

The developed environment forms a good platform for implementing and testing new IN/CC services. Developing prototypes helps the developer understand what is necessary to implement new services, what problems can be encountered, and how users experience the services.
Contents

1. INTRODUCTION
   1.1 SCOPE OF THE PROJECT 1
   1.2 PROJECT DESCRIPTION 2
   1.3 OBJECTIVE OF THE PROJECT 3
   1.4 OUTLINE OF THE REPORT 3

2. THE INTELLIGENT NETWORK
   2.1 INTRODUCTION 5
   2.2 IN CONCEPTUAL MODEL 5
      2.2.1 SERVICE PLANE 7
      2.2.2 GLOBAL FUNCTIONAL PLANE 7
      2.2.3 DISTRIBUTED FUNCTIONAL PLANE 9
      2.2.4 PHYSICAL PLANE 11
   2.3 IN SERVICES 12
      2.3.1 CAPABILITY SET 1 13
      2.3.2 CAPABILITY SET 2 14
   2.4 SUMMARY AND CONCLUSIONS 16

3. CHIPCARDS
   3.1 INTRODUCTION 17
   3.2 CHIPCARD ARCHITECTURE 18
   3.3 SECURITY 19
      3.3.1 DES 20
         3.3.1.1 DES Encryption 20
         3.3.1.2 DES Authentication 21
         3.3.1.3 Triple-DES 21
      3.3.2 RSA 21
         3.3.2.1 RSA Encryption 22
         3.3.2.2 RSA Authentication 22
      3.3.3 DIGITAL ENVELOPE 22
   3.4 ISO 7816: INTERNATIONAL STANDARD FOR CHIPCARDS 23
      3.4.1 ISO 7816-3: TRANSMISSION PROTOCOLS FOR CHIPCARDS 24
         3.4.1.1 Electrical Interface 24
         3.4.1.2 Operating Procedure 24
         3.4.1.3 Answer-to-Reset 25
         3.4.1.4 Protocol Type Selection 27
         3.4.1.5 Protocol Type T=0 28
         3.4.1.6 Protocol Type T=1 29
      3.4.2 ISO 7816-4: INTERINDUSTRY COMMANDS FOR CHIPCARDS 32
         3.4.2.1 Data Structures 32
         3.4.2.2 Security Architecture 34
         3.4.2.3 Commands and Responses 34
         3.4.2.4 Logical Channels 36
         3.4.2.5 Commands 36
3.4.2.6 Historical Bytes 36
3.4.2.7 Application-independent Card Services 38
3.5 SUMMARY AND CONCLUSIONS 38

4. CHARGING 39

4.1 INTRODUCTION 39
4.2 POST-PAYMENT 39
4.3 PRE-PAYMENT 41
4.4 SUMMARY AND CONCLUSIONS 44

5. IN/CC SERVICES 45

5.1 INTRODUCTION 45
5.2 SELECTION CRITERIA 45
5.3 SELECTION PROCESS 47
5.4 DESCRIPTION OF SELECTED IN/CC SERVICES 49
  5.4.1 SECURITY SERVICES 49
    5.4.1.1 Account Card Calling 49
    5.4.1.2 Credit Card Calling 50
    5.4.1.3 Universal Personal Telecommunications 50
    5.4.1.4 International Telecommunication Charge Card 50
  5.4.2 PROFILE SERVICES 50
    5.4.2.1 Call Distribution 50
    5.4.2.2 Abbreviated Dialing 51
    5.4.2.3 Call Rerouting Distribution 51
    5.4.2.4 Destination Call Routing 51
    5.4.2.5 Originating Call Screening 52
    5.4.2.6 Selective Call Forwarding on Busy/Don’t Answer 52
    5.4.2.7 Selective Call Forwarding 52
    5.4.2.8 Terminating Call Screening 53
    5.4.2.9 Universal Access Number 53
    5.4.2.10 Hot Line 53
    5.4.2.11 Message Store and Forward 53
  5.5 SUMMARY AND CONCLUSIONS 53

6. IN/CC ENVIRONMENT 55

6.1 INTRODUCTION 55
6.2 GENERAL ARCHITECTURE 55
6.3 PROTOTYPING ARCHITECTURE 58
6.4 COMMUNICATION PROTOCOLS 59
  6.4.1 IN APPLICATION PROTOCOL 59
  6.4.2 TCP/IP 60
6.5 ERLANG MESSAGES 62
  6.5.1.1 SCF-SSF Messages 62
  6.5.1.2 SCF-SDF Messages 63
  6.5.1.3 SCF-SRF Messages 64
6.6 SUMMARY AND CONCLUSIONS 64
7. IMPLEMENTATION

7.1 INTRODUCTION

7.2 USED TECHNOLOGY
   7.2.1 CHIPCARD TECHNOLOGY
   7.2.2 MODEM TECHNOLOGY

7.3 PROFILE MANAGEMENT SOFTWARE
   7.3.1 SERVICE ORIENTED MODE
      7.3.1.1 Service Profiles Descriptor
      7.3.1.2 Structure of transferred Service Profiles
      7.3.1.3 Structure of Service Profiles on a Chipcard
   7.3.2 USER ORIENTED MODE

7.4 IN/CC INFORMATION FLOWS

7.5 IMPLEMENTED IN/CC SERVICES
   7.5.1 ACCOUNT CARD CALLING SERVICE
   7.5.2 UNIVERSAL PERSONAL TELECOMMUNICATIONS SERVICE
   7.5.3 ABBREVIATED DIALING SERVICE

7.6 ADDITIONAL FEATURES
   7.6.1 CARD PERSONIFICATION
   7.6.2 CHARGING INFORMATION

7.7 COMMUNICATION MESSAGES
   7.7.1 PC-SRF COMMUNICATION
   7.7.2 SCF-SRF COMMUNICATION
   7.7.3 SCF-SSF COMMUNICATION

7.8 IN SOFTWARE
   7.8.1 SERVICE INDEPENDENT BUILDING BLOCKS
   7.8.2 SCRIPTS
      7.8.2.1 SCF Scripts
      7.8.2.2 SRF Script

7.9 SUMMARY AND CONCLUSIONS

8. CONCLUSIONS AND RECOMMENDATIONS

8.1 CONCLUSIONS
8.2 RECOMMENDATIONS

9. LITERATURE

APPENDIX A: JPEG COMPRESSION

APPENDIX B: STORE CHARGING10 SIB ERLANG CODE
1. Introduction

This chapter provides an introduction to this report. The area of research, a project description and the objective of the project will be discussed. To conclude this chapter an outline of the rest of the report will be presented.

1.1 Scope of the project

As a reaction to the overwhelming success of bank cards provided with a PIN code, banks, chip manufactures and telecommunication companies have been involved in the development of a variant to the PIN card, namely the smartcard or chipcard.

Chipcards are meant to be used by customers to pay electronically for relatively small amounts of money, up to about 35 guilders. These chipcards can be 'charged' with electronic money at for instance a banking office. The amount by which the card is charged is withdrawn from the user's banking account. When the customer uses the card to pay, the appropriate amount will be subtracted from the amount left on the card. In contrast with paying with PIN cards, paying with chipcards does not require a connection between the card terminal and the banking office. In other words, paying with a chipcard is an off-line process, while paying with a PIN card is an on-line process.

The advantages of an off-line paying process compared to an on-line paying process are obvious. Because no telephone connection needs to be established, the paying process takes less time and the paid party's phone bill will be reduced. Banks hope to reduce payments with pocket change and, in this way, reduce their costs. Chip manufactures hope this new market will boom and for customers paying gets easier. But what's in it for the telecommunication companies? The network operators probably will suffer from a reduction of income due to the use of chipcards instead of PIN cards.

The use of chipcards might not be as bad as it seems for telecommunication companies. Chipcards will be used for all kind of new services in the future, often in combination with telecommunication networks and services. An example is Remote Ticketing, which enables people to order, pay, and receive tickets (e.g. for the cinema) just by one telephone call.

Chipcards offer more and safer possibilities than the traditional cards with a magnetic stripe on the back. This is because chipcards contain very small computers, consisting of a microprocessor, memory, I/O logic, etc. This gives chipcards the capability to perform complex calculations (e.g. used for encryption) and to store a relatively large amount of information. Besides some information used to identify the user and the 'purse' information, that indicates the amount of electronic money left on the card, memory can be used to implement other functionality. Chipcards could be used as a train ticket, to collect Air Miles, for identification purposes, etc. And by using a chipcard terminal at home, even more services can be used, on-line.
During this project some possibilities of on-line chipcard services will be investigated. The topics on which the research will be concentrated, especially involve the Intelligent Network (IN). The IN actually is not a real network, but it is a concept, which enables service providers to develop, test and implement new, complex services in a relatively fast way. IN can be used to implement on-line chipcard services, but chipcards, in their turn, can also be used to implement IN services. Especially the latter will be the subject of this project. In other words, the project will focus on how chipcards can be used to implement IN services. In this report the intended services will be called IN/CC (IN/chipcard) services.

1.2 Project Description

As mentioned, the main subject of this project is how to use chipcards in IN services, and the following four topics will be emphasized:

- **Chipcard Technology** - To get a better view of the possibilities of chipcards, the first phase of the project will consist of studying the existing chipcard technology. Most chipcards are based on the ISO 7816 standard. Since its widely use as a lower layer standard, it seems to be useful to take a detailed look at this standard. In the Netherlands two chipcards have been introduced: the Chipknip and the Chipper. Both are based on the ISO 7816 standard. These cards will be studied to understand how chipcards are used in practice and what multi-functional capabilities they have. These multi-functional capabilities determine to what extent the Chipknip and Chipper can be used to introduce IN/CC services.

- **Charging** - People must be charged for using IN/CC services. Charging can be done in two different ways, namely pre-paid and post-paid. Post-paid means that the money is withdrawn from the user’s banking account. Up till now this procedure has been widely used to charge telephone calls and the use of IN services. Pre-paid means that the amount of money charged, is subtracted from the amount stored in his electronic purse. Since this payment method is rather new, it will be studied and described in this report. Again, to get an idea of how pre-paid transactions are processed, the implementation of this payment method in the Chipknip and the Chipper will be studied.

- **Identification and authentication** - It would be useful if a chipcard user could use his chipcard from any chipcard terminal. In other words, a user should be able to use his chipcard and related services independent from his location. To establish this, the network could read some personal information from the card, and use this information to determine who is calling and/or using a service. Chipcards may also enable more secure authentication. When all entities in a communication process are authenticated, the parties are real and can trust each other. As soon as the identity of the user is known and reliable, charging could be done from his banking account and personal information needed for services can be loaded from a central database.
• **Profile** - Some space in the chipcard’s memory might be used to store IN service related information. Flags could be used to enable or disable certain IN services. Information necessary for IN to execute a certain IN service could be stored in the chip’s memory. The UPT service is an example. The PTN could be stored on the chipcard as mentioned before. Other examples are the Account Card Calling (ACC) service (i.e. post-paid charging), and Originating Call Screening (OCS), which allows users to specify which phone numbers may be dialed. By storing a profile in the chip, the IN services related personal data does not have to be stored in a central database. This is useful when the user wants to use his service profiles in different networks from different providers. For instance, when a user is traveling a lot, but wants to use certain IN services with his personal settings and parameters independent from his location, he could upload his service profiles from his card to the respective network’s central database. Changing the profiles could be done using a PC and some profile management software. This instead of going through an extensive list of menus from a voice responding telephone system, which might not be available in your own language.

The project will consist of a theoretical part and a more practical part. Firstly, some research will be done to gather knowledge of chipcard technology and to determine how this technology can be used to implement IN/CC services. After the theoretical part of the project is finished, some ideas will be implemented and tested by using one of the existing technologies (Chipknip, Chipper or some other chipcard architecture).

### 1.3 Objective of the Project

The main objective of this project is to investigate the possibility of using chipcards to implement IN services. Chipcards are believed to become very popular and might be useful in IN services. That is why Ericsson, and especially the Intelligent Network Application Laboratory (INAL), is interested in the possibilities of chipcards in relation to IN and how certain IN/CC services can be implemented.

### 1.4 Outline of the Report

To conclude this chapter we will take a look at the rest of this report.

After the short introduction in this chapter the next chapter contains a general discussion of IN and IN services. Chapter 3 provides a general description of chipcards and highlights the most important issues of the ISO 7816 standard for chipcards. Chapter 4 gives an introduction to charging related topics. Subsequently, chapter 5 discusses IN services that can be implemented by using chipcards. In chapter 6 we will take a look at the architecture necessary to have a chipcard communicate with the IN, the protocols used for the communication between the different network entities, and the messages that may be transferred in the used prototyping environment. Chapter 7 discusses the implementation of a prototype which enables chipcard-IN interaction, with topics like the used software architecture, information flows, and implemented services. Finally, in chapter 8 some conclusions are drawn and recommendations are given.
2. The Intelligent Network

This chapter gives an introduction to the Intelligent Network (IN). It explains what IN is and why there is a need for IN. The IN Conceptual Model will be discussed, and this will provide the reader with some basic information about IN. Furthermore the IN services which are specified in the Capability Set 1 (CS-1) and in the preliminary Capability Set 2 (CS-2) will be highlighted.

2.1 Introduction

During the last few years there has been an enormous information technology push, that has influenced people’s lives both personally and professionally. As a result of the development of powerful computers and user friendly software, people are capable of doing things which hadn’t been thought of as ever being possible.

Unfortunately, the capabilities of telephone networks could not catch up with the rapid evolvement of the computer industry. The need for more sophisticated telecommunication technology increased, but telecom operators could not fulfill their customers’ demands. The main problem was that to update their infrastructure, the telecom operators needed to adapt all switches in the network. This was a slow and troublesome process and telecom operators were dependent of the cooperation of switch manufactures.

As a reaction to this problem the Intelligent Network concept was developed. The basic idea of IN is that the intelligence is moved from the switches to central computers. In this way, new services can be implemented by changing the software in these computers, whereby telecom operators are no longer dependent on switch manufactures. The interface between switches and the general purpose computers has been standardized by (among others) the International Telecommunication Union (ITU), and has been described in the ITU Q.1200 series. Besides this interface, different sets of services that IN should at least support have been described as well. These sets are called Capability Sets (CS). At this moment CS-1 has been approved and also a preliminary CS-2 is available. At the end of 1998 CS-3 is expected to be approved.

The rest of this chapter forms a brief overview of the Q.1200-series. More detailed information can be found in [Q.1200].

2.2 IN Conceptual Model

The IN Conceptual Model (INCM) is an architectural concept that can be applied to any telecommunication network. Though the standards mainly focus on the application of IN in Public Switched Telecommunication Networks (PSTN), the concept can be applied to Public Switched Packet Data Networks (PSPDN), Integrated Services Digital Networks (ISDN) and mobile networks (e.g. GSM) as well.
The INCM is depicted in Figure 2-1 and, as shown, consists of four planes. Each plane describes IN from a different point of view and at a different level of abstraction. Elements of a plane can be mapped to elements of a lower plane, thus the four planes are related to each other. The four planes are:

- **Service Plane (SP)** - Describes service aspects.
- **Global Functional Plane (GFP)** - Describes global functionality aspects.
- **Distributed Functional Plane (DFP)** - Describes distributed functionality aspects.
- **Physical Plane (PP)** - Describes physical aspects.

![Figure 2-1: Intelligent Network Conceptual Model](image-url)
The following subsections describe the four planes in more detail.

2.2.1 Service Plane

The service plane is a description of IN from a user point of view. It describes services and service features as they are perceived by the service users, without indicating how they have been implemented. A definition of the terms service and service feature is useful at this point.

- **Service** - A service is a stand-alone commercial offering, characterized by one or more core service features and optionally enhanced by other service features.

- **Service feature** - A service feature is a specific aspect of a service which is noticed by the user and which forms, alone or in conjunction with other services or service features, an service. It can be a core part of a service or an optional part offered as an enhancement to a service. A service feature is the smallest function on this plane.

The recommendation for the Service Plane should have been the contents of [Q.1202]. However, that recommendation is almost empty. The services and service features that should be supported by IN (according to CS-1) are described in [Q.1211]. Later in this chapter the CS-1 services will be discussed.

2.2.2 Global Functional Plane

The Global Functional Plane has been described in [Q.1203] and [Q.1213]. The GFP models the IN from a global, network wide point of view. Services and service features are redefined in terms of network-wide functions, which are not service or service feature specific and are therefore called Service Independent Building Blocks (SIBs).

A SIB is a reusable function, which means that it can be used for the creation of different service features. Several SIBs can be chained together to build services and service features in the Service Plane. In Figure 2-2 a graphical representation of a SIB is shown.

![Figure 2-2: Graphical representation of a SIB](image-url)
As shown in Figure 2-2 each SIB has one logical starting point and one or more logical ending points. The logical starting point is used to trigger the SIB, and the logical ending points are used to trigger the next SIBs in the chain. The SIBs are thus chained by means of their logical starting and ending points.

Also shown in the figure are two types of data parameters, namely:

- **Dynamic data parameters** - These dynamic data parameters are called Call Instance Data (CID) and exist only during the execution of a service feature. The data is specific to the call instance. CID may be input and/or output of a SIB, so data can be transferred between subsequently executed SIBs by means of common CID.

- **Static data parameters** - These static data parameters are called Service Support Data (SSD) and exist beyond the service execution. The value of SSD is fixed for all service instances and is specific to the service feature description. SSD can only be input to SIBs.

A special kind of SIB is the Basic Call Process (BCP), which provides basic call capabilities. It may receive instructions to connect or disconnect parties, or to retain a call or call attempt in the network until it can be completed. The BCP SIB thus contains the functionality of non-IN calls.

Two types of interface points between the BCP SIB and the chain of SIBs exist, namely:

- **Point of Initiation (POI)** - The POI is the BCP functional launching point from which a chain of SIBs is started.

- **Point of Return (POR)** - The POR is the functional point in the BCP where a chain of SIBs terminates.

The BCP may have more than one POI and POR, and these POIs and PORs are related to specific states in which the BCP can remain. When an IN service feature has to be executed, a POI connected to the logical starting point of the first SIB in the service chain will trigger this SIB. The chain of SIBs executes the service feature until the chain of SIBs ends at a POR. During the execution of a service feature, the BCP remains in the state in which it was at the moment of service initiation.

The order in which SIBs are chained together to accomplish services is defined by the Global Service Logic (GSL). The GSL can be considered to be the 'glue' that keeps the SIBs (including the BCP SIB) together. Figure 2-3 illustrates how SIBs can be chained together by the Global Service Logic. This figure also shows how a service feature is launched from a POI in the BCP, and how the service feature ends at a POR.
2.2.3 Distributed Functional Plane

In [Q.1204] and [Q.1214] the Distributed Functional Plane (DFP) is described. By specifying the elements of the GFP in terms of elements of the DFP, the distribution aspects of the elements of the GFP become clear. The distribution of the GFP is shown in the IN Functional Model (INFM), which illustrates the grouping of functions, their relations and their actions. Figure 2-4 depicts the IN Functional Model.

The ovals in Figure 2-4 represent Functional Entities (FEs), which can be seen as unique groups of functions in a single location and subsets of the total set of functions required for providing a service. The SIBs of the GFP are modeled in the DFP by a sequence of particular Functional Entity Actions (FEAs) performed in the FEs.
The IN Functional Model also illustrates how different FEs may interact with each other during the execution of services or service features. These interactions between communicating FEs are called Information Flows (IFs), and are depicted as lines between FEs in Figure 2-4.

The different FEs that have been defined are:

- **Call Control Agent Function (CCAF)** - The CCAF provides access for users. It is the interface between the user and the network Call Control Functions.

- **Call Control Function (CCF)** - The CCF provides call/connection processing and control.

- **Service Switching Function (SSF)** - The SSF invokes IN service processing.

- **Service Control Function (SCF)** - The SCF controls the processing of IN services by executing a service logic program.

- **Service Data Function (SDF)** - The SDF contains customer and network data for real-time access by the SCF.

- **Specialized Resource Function (SRF)** - The SRF provides specialized resources required for interaction with the user during the execution of IN services (e.g., announcements, speech recognition and synthesis, DTMF sending and receiving).

- **Service Creation Environment Function (SCEF)** - The SCEF allows IN services to be defined, developed, tested and input to the Service Management Function.

- **Service Management Agent Function (SMAF)** - The SMAF provides an interface between service managers and the Service Management Function.

- **Service Management Function (SMF)** - The SMF allows deployment and provision of IN services and allows the support of the ongoing operation.

The CCAF and the CCF are related to the normal call processing (i.e., non-IN processing). The other FEs are never involved in a normal telephone call. When the SSF recognizes that IN call processing is required, normal call processing is suspended and call control is given to the SCF, which executes a service program. The SCF can, for instance, request the SDF for a list of numbers from its database or request the SRF to play an announcement to the user. When the SCF has completed the service program, call control is returned to the SSF.

The SCEF, SMAF and SMF are not related to call processing, but to the creation, testing and management of IN services.
2.2.4 Physical Plane

The Physical Plane (PP) (described in [Q.1205] and [Q.1215]) models the physical aspects of IN networks. The model identifies the different Physical Entities (PEs), or physical nodes, and the interfaces between those entities.

An FE must be found in at least one PE, but more than one FE can form a PE. However, there can never be more than one FE of the same type in one PE, and it is not possible to divide an FE between two PEs.

Figure 2-5 shows a scenario for physical architectures.
Examples of PEs, as shown in Figure 2-5, are:

- **Service Switching Point (SSP)** - Contains the core PEs CCF and SSF, and optional the SCF, SRF, CCAF and/or the SDF.
- **Service Node (SN)** - Contains the CCF, SSF, SCF, SDF and SRF.
- **Intelligent Peripheral (IP)** - Contains the SRF.
- **Network Access Point (NAP)** - Contains at least the CCF and optional the CCAF.
- **Service Control Point (SCP)** - Contains the SCF and SDF and is connected to SSPs by a signaling network.
- **Adjunct (AD)** - Also contains the SCF and SDF but is, in contrast with the SCP, directly connected to SSPs.

Two more examples of PEs, not shown in Figure 2-5, are:

- **Service Switching and Control Point (SSCP)** - The SSCP is the SCP and the SSP combined in a single node. This node may also contain the SRF.
- **Service Data Point (SDP)** - Contains the SDF.

Communication between two PEs is implemented in the PP through the Intelligent Network Application Protocol (INAP), a component layer in the Transaction Capability Application Part (TCAP), which is a standardized OSI-based application layer protocol. The PEs communicate by operations, which are messages exchanged between them. These messages can be exchanged by using existing protocols like ITU's Signaling System No. 7 (SS7) at the Network Node Interface (NNI) or DSSI at the User Network Interface (UNI).

The next paragraph will discuss IN services that have been defined in CS-1 and CS-2.

### 2.3 IN Services

The introduction of IN in existing telecommunication networks, is a very slow process. This makes the standardization of IN a difficult task. There is a great change that, before a standard has been implemented, the technological possibilities and the requirements of customers have been changed, and thus making the standard old-fashion. To take full advantage of the technology at a given point of time, it is necessary to define specific phases in the standardization process. This concept of a phased evolution to a target architecture is used for IN. The different phases are related to the different Capability Sets (CSs). Up till now, only CS-1 has been defined, and a preliminary CS-2 is available. The following subparagraphs will discuss the services that have been specified in these CSs.
2.3.1 Capability Set 1

The Capability Set 1 (CS-1) is defined in the Q.121x series [Q.1210]. CS-1 includes SIBs (refer to [Q.1213]), service features, and services (refer to [Q.1211]). The defined SIBs and service features represent a minimum set required to define the CS-1 targeted services. The SIBs and service features will not be discussed here, but the defined services are described subsequently. Table 2-1 shows the IN services which have been defined in CS-1.

<table>
<thead>
<tr>
<th>IN Service</th>
<th>Abbr.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abbreviated Dialing</td>
<td>ABD</td>
<td>Allows business subscribers to use abbreviated phone numbers, consisting of, for instance, only four numbers, to call others in their company.</td>
</tr>
<tr>
<td>Account Card Calling</td>
<td>ACC</td>
<td>Allows subscribers to call from any terminal (with or without a card terminal) and have the call charged to a personal or business account as indicated by the card content, rather than to the terminal’s owner.</td>
</tr>
<tr>
<td>Automatic Alternative Billing</td>
<td>AAB</td>
<td>Allows subscribers to call from any terminal and have the call charged to his own personal account or ask the called party to accept the costs of the phone call.</td>
</tr>
<tr>
<td>Call Distribution</td>
<td>CD</td>
<td>Routes incoming calls to different destinations, according to a distribution algorithm, which may be managed by the subscriber. The three types of distribution algorithms are: circular, percentage and hierarchical distribution.</td>
</tr>
<tr>
<td>Call Forwarding</td>
<td>CF</td>
<td>Forwards calls to another number. The service is managed by the subscriber.</td>
</tr>
<tr>
<td>Call Rerouting Distribution</td>
<td>CRD</td>
<td>Allows subscribers to have incoming calls encountering a triggering condition (busy, specified number of rings, queue overload or call limiter) rerouted according to a predefined choice (to another number, on an announcement, or queued).</td>
</tr>
<tr>
<td>Completion of Calls to Busy Subscriber</td>
<td>CCBS</td>
<td>Allows the calling party encountering a busy destination to be informed when the destination becomes free.</td>
</tr>
<tr>
<td>Conference Calling</td>
<td>CON</td>
<td>Connects multiple parties in a single conversation.</td>
</tr>
<tr>
<td>Credit Card Calling</td>
<td>CCC</td>
<td>Charges calls made from any access point to any destination to a specified account.</td>
</tr>
<tr>
<td>Destination Call Routing</td>
<td>DCR</td>
<td>Routes calls to destinations according to criteria like time, date, originating area, calling line identity, priority, proportional routing, etc.</td>
</tr>
<tr>
<td>Follow-Me Diversion</td>
<td>FMD</td>
<td>Allows users to remotely control forwarding capabilities of their primary telephone number from any terminal in the network.</td>
</tr>
<tr>
<td>Freephone</td>
<td>FPH</td>
<td>Allows reverse charging. The called party accepts the whole cost of the call (e.g. 0800 numbers).</td>
</tr>
<tr>
<td>Malicious Call Identification</td>
<td>MCI</td>
<td>Enables subscribers to control the logging of malicious calls that are received. At least the called party number, the calling party number and the time and data will be registered.</td>
</tr>
<tr>
<td>Mass Calling</td>
<td>MAS</td>
<td>Allows the network operator to temporarily allocate a single directory number to the served user. When an end user calls the number, announcements will be played and by entering digits, which will be registered, the user can make choices. The served user will be supplied with the total 'votes' for each preference.</td>
</tr>
<tr>
<td>Originating Call Screening</td>
<td>OCS</td>
<td>Specifies which outgoing calls are or are not allowed. If an outgoing call is not allowed, this can be overridden by anyone with the proper identity code.</td>
</tr>
</tbody>
</table>
2.3.2 Capability Set 2

The Capability Set 2 (CS-2) is defined in the Q.122x series [Q.1220], and is a superset of CS-1. As CS-1, CS-2 also includes SIBs (refer to [Q.1223]), service features, and services (refer to [Q.1221]). Again the defined SIBs and service features represent a minimum set required to define the CS-2 targeted services. The defined services are described in this subparagraph.

In CS-2 three types of services have been defined, namely telecommunication services, service management services and service creation services. However, this subsection only discusses the telecommunication services. The other two types are beyond the scope of this report.

Table 2-2 shows the IN services which have been defined in CS-2.

<table>
<thead>
<tr>
<th>IN Service</th>
<th>Abbr.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Premium Rate</td>
<td>PRM</td>
<td>Allows to pay the called party, considered as an added value service provider, for its service. This is done by charging calls at special, higher rates. The network operator shares the revenues with the service provider.</td>
</tr>
<tr>
<td>Security Screening</td>
<td>SEC</td>
<td>Attempts of users to access a subscriber's network, system or application will be screened. Call data and invalid codes attempted are recorded.</td>
</tr>
<tr>
<td>Selective Call Forwarding on busyno answer</td>
<td>SCF-BY/DA</td>
<td>Allows subscribers to forward certain pre-selected calls if the called user is busy or does not answer within Y seconds or X rings.</td>
</tr>
<tr>
<td>Selective Call Forwarding</td>
<td>SCF</td>
<td>Allows called parties to forward pre-selected calls, no matter what the status of the called party line is.</td>
</tr>
<tr>
<td>Call Forwarding on Busy</td>
<td>CFBY</td>
<td>Allows subscribers to have incoming calls forwarded to another number if a busy condition is encountered.</td>
</tr>
<tr>
<td>Call Forwarding on Don't Answer</td>
<td>CFDA</td>
<td>Allows subscribers to have incoming calls forwarded to another number if a don't answer condition is encountered.</td>
</tr>
<tr>
<td>Split Charging</td>
<td>SPL</td>
<td>Enables a network operator to split the charges for a call. The calling and the called party are each being charged for one part of the call.</td>
</tr>
<tr>
<td>Televoting</td>
<td>VOT</td>
<td>Enables subscribers to vote by phone. To indicate their vote, users call one of two or more numbers or call a unique number and, after prompting, give their choice by dialing a number or by voice.</td>
</tr>
<tr>
<td>Terminating Call Screening</td>
<td>TCS</td>
<td>Specifies which incoming calls are or are not allowed, according to a screening list, and optionally, by the time of day.</td>
</tr>
<tr>
<td>Universal Access Number</td>
<td>UAN</td>
<td>Allows a subscriber with several terminating lines to be reached with a unique directory number. Incoming calls are routed to terminating lines, based upon the area the call originated. The exact routing may be specified by the subscriber.</td>
</tr>
<tr>
<td>Universal Personal Telecommunications</td>
<td>UPT</td>
<td>Allows users to access any type of service and receive calls on the basis of a unique Personal Telecommunication Number (PTN) at any network access across multiple networks.</td>
</tr>
<tr>
<td>User-Defined Routing</td>
<td>UDR</td>
<td>Allows subscribers to specify how outgoing calls have to be routed, either through private, public or virtual facilities (or a mix of facilities).</td>
</tr>
<tr>
<td>Virtual Private Network</td>
<td>VPN</td>
<td>Allows users to build a private network (a virtual PABX) with public network resources. Private network capabilities like private numbering plan (PNP), call transfer and call hold can be used.</td>
</tr>
</tbody>
</table>
### Table 2-2: IN services defined in CS-2

<table>
<thead>
<tr>
<th>IN Service</th>
<th>Abbr.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call Hold</td>
<td>CH</td>
<td>Allows a subscriber to place a call on hold, play an announcement to the held party, and initiate a new call. The user can subsequently resume the original call.</td>
</tr>
<tr>
<td>Call Transfer</td>
<td>TRA</td>
<td>Allows the subscriber to hold an incoming call, dial a new destination number and connect the held party to the new destination by releasing his own connection.</td>
</tr>
<tr>
<td>Call Waiting</td>
<td>CW</td>
<td>Allows a subscriber to hold an active call, when a new incoming call is detected, and to accept this new call. The subscriber is then connected to two parties and can toggle between them.</td>
</tr>
<tr>
<td>Completion of Call to Busy Subscriber</td>
<td>CCBS</td>
<td>As defined in CS-1.</td>
</tr>
<tr>
<td>Conference Calling</td>
<td>CONF</td>
<td>As defined in CS-1.</td>
</tr>
<tr>
<td>Future Public Land Mobile Telecommunication System</td>
<td>PPL</td>
<td>Provides a link between a mobile terminal and a fixed network access point, enabling the user to move freely during the use of telecommunication services. The network is able to locate and identify terminals within a network domain, whether or not the user is making a call.</td>
</tr>
<tr>
<td></td>
<td>MTS</td>
<td></td>
</tr>
<tr>
<td>Global Virtual Network Service</td>
<td>GVNS</td>
<td>This is a global switched Virtual Private Network service, as defined in CS-1, supported by multiple networks.</td>
</tr>
<tr>
<td>Hot Line</td>
<td>HOT</td>
<td>Allows users to initiate a call, without providing the called party information required by the network to route the call. The routing information is stored in the network by prior subscription. Incoming calls are screened and only allowed for specified parties.</td>
</tr>
<tr>
<td>International Telecommunication Charge Card</td>
<td>ITCC</td>
<td>Allows a telecommunication charge card holder to make use of telecommunication services provided by the card acceptor and have the charges billed by the card issuer to the customer's account number.</td>
</tr>
<tr>
<td>Internetwork Freephone</td>
<td>IFPH</td>
<td>Allows a subscriber to have one or more installations to be reached from a specific network other than his network with a Freephone number, and to be charged for the call.</td>
</tr>
<tr>
<td>Internetwork Mass Calling</td>
<td>IMAS</td>
<td>Allows the Mass Calling service as defined in CS-1 to be used with the originating call in one network and the terminating call in another network.</td>
</tr>
<tr>
<td>Internetwork Premium Rate</td>
<td>IPRM</td>
<td>Allows to pay the called party, considered as an added value service provider, for its service. Calls are charged at a higher rate. The communication is between callers in one network and the service provider in another network.</td>
</tr>
<tr>
<td>Internetwork Televoting</td>
<td>IVOT</td>
<td>Allows the Televoting Service as defined in CS-1 to be used by callers in one network and a service provider in another network.</td>
</tr>
<tr>
<td>Message Store and Forward</td>
<td>MSF</td>
<td>Allows users to send a message (e.g. voice, data or fax) to one or more destinations. Different methods and/or times of delivery may be specified.</td>
</tr>
<tr>
<td>Multimedia</td>
<td>MMD</td>
<td>Allows users to send or receive a mixture of voice, data, image and video information. Delivery of information from disparate sources can be synchronized and controlled by the user. Delivery from one source to multiple recipients and from multiple sources to a single recipient is possible. During a call other types of information streams may be added to or removed from the call.</td>
</tr>
<tr>
<td>Terminating Key Code Screening</td>
<td>TKCS</td>
<td>Allows a user to screen incoming calls by means of a user defined key, which must be entered by callers to get a connection to the called party.</td>
</tr>
<tr>
<td>Universal Personal Telecommunications</td>
<td>UPT</td>
<td>As defined in CS-1.</td>
</tr>
</tbody>
</table>
2.4 Summary and Conclusions

This chapter provided an introduction to the Intelligent Network concept. IN enables network operators to quickly react to customers’ needs for new services. The IN Conceptual Model describes four planes, which each provide a different point of view to IN. The four planes are the Service Plane, the Global Functional Plane, the Distributed Functional Plane, and the Physical Plane.

The standardization and implementation of IN have been divided into different phases. These phases are called Capability Sets (CSs). Up till now, only CS-1 has been specified, and a preliminary version of CS-2 is available. The defined services in these CSs have been described briefly.
3. Chipcards

This chapter describes general aspects of chipcards. After a short introduction, a general chipcard architecture, cryptographic methods, and the ISO 7816 standard for chipcards will be discussed. This standard is the basis for almost any chipcard available and specifies card and card terminal related characteristics.

3.1 Introduction

These days there is an increasing interest in chipcards. The capabilities of chipcards seem to be enormous. All kinds of information can be stored safely on a chipcard. In the near future just a few chipcards might replace one’s credit cards, pocket change, passport, driver’s license, medical documents, telephone card, public transportation card, etc. Chipcards also seem to be the key in making transmission of secret information through a network safe, which implies that, for instance, secure remote financial transactions will become feasible.

A chipcard has the size of a credit card and is equipped with an IC with a microprocessor, memory and I/O logic. In contrast with the traditional cards with magnetic stripes, chipcards can perform cryptographic calculations and handshake protocols. Information exchange between a chipcard and its environment can be done in a safer way.

Advanced cryptographic functions like Public Key Encryption (PKE)\(^1\) are provided and offer two solutions. First, by sending data encrypted over the network, no unauthorized people can read the data. Secondly, encrypted data can be used as a digital signature, ensuring that the sender is authorized.

The basis of most chipcards is the ISO 7816 standard. This standard specifies the physical and electrical characteristics of chipcards and terminals, and the transmission protocols used in chipcard-terminal communication. Also the two Dutch chipcards, the Chipknip and the Chipper, comply with the ISO 7816 standard. However, this does not mean that these chipcards are exactly alike. The microprocessors and operating systems used, are different. Also different applications have been implemented.

This chapter describes some general aspects of chipcards. The Chipknip and the Chipper have been studied, but they will not be described in this report. Most information about the two Dutch chipcards is classified and may not be made public. Therefor the Chipknip and the Chipper have been described in a separate classified document (HELV97), which will only be distributed to those who are entitled to get this information.

---

\(^1\) An example of PKE is the RSA algorithm, which will be discussed in section 3.3.2. More general information about PKE can for instance be found on the internet: http://uep.ams.ameslab.gov/dev/charlie/pke.save/pke.html
3.2 Chipcard Architecture

This paragraph discusses the architecture of chipcards. Figure 3-1 shows a typical chipcard architecture.

As shown in Figure 3-1, a chipcard has six I/O pins. The functions of the pins are explained in Table 3-1.

<table>
<thead>
<tr>
<th>Pin</th>
<th>I/O</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLK</td>
<td>I</td>
<td>Provides an external clock signal to the chipcard.</td>
</tr>
<tr>
<td>RST</td>
<td>I</td>
<td>Used to reset the chipcard.</td>
</tr>
<tr>
<td>VPP</td>
<td>I</td>
<td>This optional pin provides the chipcard with the programming voltage, used to program or erase the EEPROM.</td>
</tr>
<tr>
<td>VCC</td>
<td>I</td>
<td>Power supply for the chipcard.</td>
</tr>
<tr>
<td>GND</td>
<td>I</td>
<td>Ground, serves as a reference signal.</td>
</tr>
<tr>
<td>I/O</td>
<td>I/O</td>
<td>This is a serial I/O pin used to enable communication between the chipcard and its environment (e.g. terminal).</td>
</tr>
</tbody>
</table>
As shown in Figure 3-1, the main parts of a chipcard are:

- **Central Processing Unit (CPU)** - Performs calculations, interprets and performs commands, and executes applications. Typically the CPU has an 8-bit data bus, address bus and registers, and works with a clock frequency in the range from 3 to 5 MHz.

- **ROM** - Is used to store the operating system, fixed data and standard routines. Data stored in ROM cannot be changed and will still be available after power supply has been removed. Typically, ROM consists of up to 20 Kbytes.

- **RAM** - Is used by the CPU as a stack memory to store results from calculations. When power supply is removed from the card, all data in RAM will be lost. Typically, RAM consists of a maximum of 780 bytes.

- **EEPROM** - This is Electrically Erasable Programmable ROM, and is used to store data that is not fixed, but might be changed, and that will still be available after power supply has been removed. Users can use EEPROM to store personal and application related information. Typically, EEPROM consists of 1 to 16 Kbytes.

- **I/O logic** - Makes communication with the external world (e.g. a card terminal) possible. The I/O logic implements half-duplex serial communication. Incoming and outgoing data may be stored in buffers temporarily. When data is coming in, the I/O logic will alert the CPU by asserting an interrupt.

- **Random Number Generator** - Generates random numbers, which are used for security related functions. More information about security can be found in the next paragraph.

The presented architecture does not provide an in-depth view of chipcards, but roughly shows a general chipcard's architecture. A lot of chip manufactures develop chipcards, which all have different architectures, but more or less look like the architecture presented in this paragraph.

The next section discusses some security related topics.

### 3.3 Security

A very important aspect of the chipcard technology is security. Information stored on a chipcard will often be personal, secret information, which should not be accessible by unauthorized parties. Security is also very important if a chipcard is used as an electronic purse. It should be impossible to change the purse information (e.g. increasing the amount of money stored on the card) and to emulate a charged chipcard. Also emulating terminals or hosts in a telecommunication network should be impossible. Further, confidential information transferred between a card and a terminal or host, should not be readable by unauthorized parties.
To achieve secure use of chipcards, the following security mechanisms are used:

- **Encryption** - Is used to make transferred data unreadable to unauthorized parties. By ciphering the data before transmission and deciphering the data after reception, the data transfer will be secure. Unauthorized parties might have been able to filter the transmitted data, but they will not be able to decipher the data without the secret key. If an unauthorized party has intercepted and changed the transferred data, the receiver will notice this.

- **Authentication** - Is used to ascertain that the chipcard, terminal and/or host are real, not emulated, and authorized entities. By sending a digital signature one of the entities can prove to other entities that it can be trusted. Only if all involved entities have proved their reliability to all other entities, secure transfer of data is ensured.

The two most important cryptographic methods to implement encryption and authentication are DES and RSA. These methods are the subject of the following subparagraphs. Also a combination of DES and RSA will be discussed.

### 3.3.1 DES

The Data Encryption Standard (DES) is the most used cryptographic method in the chipcard technology. It is a symmetric method, which means that the sending and receiving device contain the same secret key to both encode and decode the data. Figure 3-2 illustrates how DES works.

![Figure 3-2: DES encryption](image)

#### 3.3.1.1 DES Encryption

Sender A encrypts the original message M using a secret key K, resulting in the encoded message M'. This message is sent to receiver B, which decodes M' using the same secret key. Finally, B obtains the original message M.
3.3.1.2 DES Authentication

So DES can be used to encrypt messages, but it can be used for authentication purposes as well. Suppose the chipcard wants to authenticate the terminal. The chipcard generates a random number R, encrypts it with key K and transmits the resulting R'. The terminal receives R' and uses its key K to decrypt it. If the terminal used the proper key, the result will be the original random number R. The result is sent back to the chipcard, which compares it with the original number R. If both numbers are equal, then the chipcard knows it can trust the terminal. The terminal can use the same method to authorize the card.

3.3.1.3 Triple-DES

The DES algorithm can be used with a double length key. In this case, at the transmission side, data is encoded with the first half of the key, decoded with the second half, and again encoded with the first half of the key. At the reception side, decoding of the data is performed using the same three steps. This algorithm is called triple-DES.

The main disadvantage of DES is that the secret key is shared among all involved devices. The more widely the key is used, the weaker the security gets. An asymmetric cryptographic method like RSA solves this problem.

3.3.2 RSA

The Rivest, Shamir, Adleman (RSA) cryptographic method is asymmetric. This means that different keys are used to encode and decode messages. Both the receiver and the sender contain a private key and at least one public key. If the sender encodes a message with the public key K₁, it can only be decoded by the receiver with its corresponding private key K₂. If the sender encodes a message with its private key K, it can only be decoded by the receiver with the corresponding public key K₂. This is shown in Figure 3-3.

![Figure 3-3: RSA encryption](image_url)

The RSA method can be used to encrypt messages, and for authentication purposes.
3.3.2.1 RSA Encryption

Suppose A wants to send a message M to B. A encodes M resulting in M', by using B's public key. To decode M', B uses its private key and obtains the original message M. The relationship between B's public key and B's private key ensures that B correctly recovers M. Since only B knows its private key, only B can decode M'.

3.3.2.2 RSA Authentication

Suppose A wants to authenticate B. First A generates a random number R and sends it to B. Then B uses its private key to encode R, and sends the resulting R' back to A. Finally, A uses B's public key to decode R', and compares the result with the original random number R. If both numbers are equal, A knows that B can be trusted.

A disadvantage of RSA is that it is a much slower cryptographic method than DES. The next subparagraph discusses a combination of DES and RSA that utilizes the advantages of both methods, and reduces the disadvantages.

3.3.3 Digital Envelope

In practice, RSA will be used in combination with a symmetric cryptographic method like DES. The reason for this is, that RSA is too slow to encrypt large messages. By encrypting the message using DES and encrypting the DES key using RSA, a relatively fast and safe method is obtained. This method is called digital envelope, and Figure 3-4 illustrates how it works.

![Digital Envelope Diagram]

*Figure 3-4: Digital envelope*
Message $M$ is encrypted using a DES key $K$. The key $K$ itself is encrypted using $B$'s public key $K_1$. The resulting $M'$ and $K'$ together form the digital envelope, which is sent to $B$. Only $B$ can decrypt $K'$ with its private key $K_2$. The DES key $K$ is obtained and is used to decode $M'$.

The next paragraph will describe the ISO 7816 standard. This standard is very important, because it is the basis for most chipcards.

### 3.4 ISO 7816: International Standard for Chipcards

The International Organization for Standardization (ISO) has defined the ISO 7816 standard, 'Identification cards - Integrated circuit(s) cards with contacts'. This paragraph provides a brief description of the standard.

Figure 3-5 shows how the different parts of the ISO 7816 standard are related to the OSI Reference Model.

<table>
<thead>
<tr>
<th>OSI Reference Model</th>
<th>ISO 7816 Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Apnl.</td>
</tr>
<tr>
<td>Presentation</td>
<td>General functions and information ISO 7816-5/6</td>
</tr>
<tr>
<td>Session</td>
<td>Inter-industry command set ISO 7816-4</td>
</tr>
<tr>
<td>Transport</td>
<td>Character protocol $T=0$ ISO 7816-3 Paragraph 6</td>
</tr>
<tr>
<td>Network</td>
<td>Block protocol $T=1$ ISO 7816-3 Paragraph 9</td>
</tr>
<tr>
<td>Data link</td>
<td>Electrical characteristics ISO 7816-2/3</td>
</tr>
<tr>
<td>Physical</td>
<td>Physical characteristics ISO 7816-1</td>
</tr>
</tbody>
</table>

*Figure 3-5: Relation between the OSI Reference Model and the ISO 7816 standard*

As shown in Figure 3-5, the standard consists of six parts. The first two parts describe the physical characteristics and the dimensions and locations of the contacts. Part 5 of ISO 7816 is called: 'Numbering system and registration procedure for application identifiers'. It specifies a system to number application identifiers and a registration procedure for application provider identifiers. Part 6, 'Interindustry data elements', describes the characteristics of data elements. However, the information provided in these parts is beyond the scope of this report and will not be further discussed. For more information refer to [ISO7816-1], [ISO7816-2], [ISO7816-5] and [ISO7816-6].

The following sections describes part 3 and 4 of the ISO 7816 standard.
3.4.1 ISO 7816-3: Transmission Protocols for Chipcards

This paragraph discusses part 3 of the ISO 7816 standard, as specified in [ISO7816-3]. It describes electronic signals and transmission protocols.

3.4.1.1 Electrical Interface

As described before, for chipcards six I/O pins have been defined: CLK, RST, VPP, VCC, GND and I/O. For these pins, four different states are possible (as defined in [ISO1177]):

- State H - Logical high level
- State L - Logical low level
- State Z - Mark
- State A - Space

3.4.1.2 Operating Procedure

The interface device (IFD) to which the chipcard is electrically connected, communicates with the chipcard through the following operations:

- **Connection and activation** of the contacts by the IFD. Activation of the contacts consists of the following consecutive operations:
  - RST in state L
  - VCC will be powered
  - I/O will be put in reception mode
  - VPP is raised to idle state
  - CLK will be provided with a stable clock signal.

- **Reset** of the card. Three different kinds of card resets have been defined:
  - *Asynchronous internally reset* - The card performs a self-reset after a few cycles of the clock signal.
  - *Asynchronous active low reset* - The card is reset by maintaining RST low for at least 40000 clock cycles after the clock signal is applied to CLK.
  - *Synchronous active high reset* - When a synchronous answer of the card is expected, CLK remains low until RST has become high. Then the clock signal is activated and RST goes low right after CLK has gone low. During the next clock cycle the first data bit of the Answer-to-Reset is sent on I/O.

- **Information exchange** between the card and the IFD, always initiated by an Answer-to-Reset (ATR) sent by the card. The next subsection will describe the ATR. This is followed by some more subsections related to the information exchange between the card and the IFD.
**Deactivation** of the contacts by the IFD. Deactivation of the contacts consists of the following consecutive operations:

- RST in state L
- CLK in state L
- VPP will become inactive
- I/O will go in state A
- VCC will become inactive

### 3.4.1.3 Answer-to-Reset

For the communication between chipcards and IFDs, two types of transmission are defined: asynchronous and synchronous transmission. We will consider both types.

**Asynchronous Transmission**

In asynchronous transmission, characters are transmitted on the I/O line in an asynchronous half-duplex mode. A character is defined as ten bits: a start bit in state A, eight bits of information, and a bit for even parity checking. A reception of a character is perceived by the IFD by detecting a transition on the I/O pin from the Z state to the A state. Between each character there is a certain guardtime, during which the I/O pin should be in the Z state again. The transmitter tests the I/O pin during the guardtime. If I/O is in the Z state, the transmission of the character can be assumed to be correct. If I/O is in the A state, the transmission was incorrect, and the transmitter will retransmit the character.

The structure of the ATR is as shown in Figure 3-6 on the next page. The initial character TS always has one of the following structures:

- **Inverse convention:** \text{AZZAAAAAAAZ}
  - Indicates that state A is the logic level '1' and that data is transmitted in most significant bit (msb) first mode.

- **Direct convention:** \text{AZZAZZZAAZ}
  - Indicates that state Z is the logical level '1' and that data is transmitted in least significant bit (lsb) first mode.

The first four bits (AZZA) form a synchronization sequence. The next three bits (AAA or ZZZ) indicate whether the inverse or direct convention (refer to [ISO1177]) is being used, and the last three bits (AAZ) are used to check parity. The initial character also indicates the data rate that the card uses to communicate with the IFD.

The format character T0 is mandatory and contains two parts:

- **Y** - This is the most significant half byte (b5 to b8) and indicates whether or not subsequent characters TA1 TB1 TC1 TD1 are present. For instance, if b5 is '1' then TA1 will be transmitted.

- **K** - This is the least significant half byte (b4 to b1) and indicates the number (0 to 15) of historical characters that will be transmitted.
As shown in Figure 3-6 the interface characters code parameters FI, DI, II, PI1, PI2 and N. FI and DI indicate the data rate that will be used in subsequent transmissions. II indicates the maximum programming current, and PI1 and PI2 indicate the maximum programming voltage VPP. Finally, N indicates that the card requires some extra guardtime.
Character TD\(_i\) codes Y\(_{i+1}\) and T. The coding of Y\(_i\) is equal to the coding of Y\(_1\) and indicates whether or not TA\(_i\), TB\(_i\) and/or TC\(_i\) will be transmitted. Parameter T indicates which of the following protocol types is used:

- **T=0** - The asynchronous half duplex character transmission protocol.
- **T=1** - The asynchronous half duplex block transmission protocol.
- **T=2..15** - Reserved for future use.

These protocols will be described later in this subsection.

After the ATR, the card is in one of the following two modes:

- **Negotiable mode** - In this mode, the default values of parameters Fl and DI used during ATR and the first offered protocol type in ATR will be used until a successful Protocol Type Selection (PTS) procedure is performed. The PTS procedure will be discussed later.

- **Specific mode** - In this mode, the parameters Fl and DI and the protocol type required in the ATR will apply directly after the ATR.

If TA\(_2\) is present, the specific mode is selected. If TA\(_2\) is not present the negotiable mode is present.

The use of the historical characters is not defined in the ISO 7816-3 standard, but in ISO 7816-4. We will discuss the historical characters in the next subparagraph.

The check character TCK will be such that the exclusive-oring of all bytes from TO to TCK included is null, and is used to check the correct transmission of the ATR.

**Synchronous Transmission**

In synchronous transmission a linear relation between the bit rate on the I/O line and the clock frequency on CLK exists. For instance, a clock frequency of 50 kHz implies a bit rate of 50 kbit/s. Data bits can be sampled on the rising edge of CLK.

After reset, the card answers with an ATR, containing a 32 bits header. The least significant bit is transmitted first. The header starts with H1 and H2, both 8 bit fields. H1 codes the protocol type to be used, and H2 codes parameters for the protocol type coded in H1. The ISO 7816-3 standard does not contain more detailed information about the contents of the header.

**3.4.1.4 Protocol Type Selection**

If more than one protocol type is implemented in the card, it will indicate which protocols it supports in different TD\(_i\) characters. If the T=0 protocol is not implemented and Protocol Type Selection (PTS) is not supported, the IFD will reject or reset the card. If the card supports T=0, and indicated this in TD\(_i\) as the first offered protocol, then T=0 is assumed to be the protocol used in subsequent communication, unless PTS is performed.
If PTS is performed, the IFD will select, from the different protocols implemented in the card, which protocol to use, and informs the card about its selection. From that point both parties know what protocol will be used, and subsequent transmissions will indeed be conform the selected protocol type. The following subsections describe the two defined protocol types.

3.4.1.5 Protocol Type T=0

Protocol type T=0 is an asynchronous half duplex character transmission protocol. All data is transmitted as characters. A character consists of, as described before, a start bit, eight data bits and an even parity bit. If the receiver detects an error (i.e. a parity error), then the character will be retransmitted.

The IFD always initiates communication by sending a command to the card. The command tells the card what to do, and consequently the card answers by sending procedure bytes. Dependent on the procedure bytes, transfer of data bytes is allowed or not. It is assumed that the direction of data transfer is known from the command sent by the IFD. In other words, some commands initiate incoming data transfers (to the card), others initiate outgoing transfers (from the card).

The command header consists of five successive bytes:

- **CLA** - Codes the instruction class. ‘FF’ is reserved for PTS.
- **INS** - This is the instruction code in the instruction class.
- **P1** - This is a reference completing the instruction code (e.g. an address).
- **P2** - This is also a reference completing the instruction code.
- **P3** - Codes the number of data bytes to be transferred during execution of the command.

In an outgoing data transfer, P3=0 indicates a 256-byte data transfer. In an incoming data transfer, P3=0 indicates that no data has to be transferred.

The chipcard controls the subsequent data transfer by sending procedure bytes to the IFD. Three types of procedure bytes are specified: ACK, NULL and SW1. The values and results of these procedure bytes are shown in Table 3-2.

<table>
<thead>
<tr>
<th>Byte</th>
<th>Value</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK</td>
<td>INS</td>
<td>VPP is idle. All remaining data bytes are transferred subsequently.</td>
</tr>
<tr>
<td></td>
<td>INS+1</td>
<td>VPP is active. All remaining data bytes are transferred subsequently.</td>
</tr>
<tr>
<td></td>
<td>not (INS)</td>
<td>VPP is idle. Next data byte is transferred subsequently.</td>
</tr>
<tr>
<td></td>
<td>not (INS+1)</td>
<td>VPP is active. Next data byte is transferred subsequently.</td>
</tr>
<tr>
<td>NULL</td>
<td>'60'</td>
<td>No further action on VPP. The IFD waits for a new procedure byte.</td>
</tr>
<tr>
<td>SW1</td>
<td>SW1</td>
<td>VPP is idle. The IFD waits for a SW2 byte.</td>
</tr>
</tbody>
</table>

The sequence SW1-SW2 gives the status of the card. SW1 is coded by ‘6x’ or ‘9x’, except for ‘60’. SW2 can have any value.
The normal ending of a data transfer is indicated by SW1-SW2 = ‘90’-‘00’. The meaning of some other values of SW1 have been defined as shown in Table 3-3.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘6F’</td>
<td>No precise diagnosis is given.</td>
</tr>
<tr>
<td>‘6E’</td>
<td>The card does not support the instruction.</td>
</tr>
<tr>
<td>‘6D’</td>
<td>The instruction code is not programmed or is invalid.</td>
</tr>
<tr>
<td>‘6B’</td>
<td>The reference is incorrect.</td>
</tr>
<tr>
<td>‘67’</td>
<td>The length is incorrect.</td>
</tr>
</tbody>
</table>

Other values have not been defined in ISO 7816 and are application specific.

3.4.1.6 Protocol Type T=1

Protocol type T=1 is defined in Amendment 1 of the ISO 7816-3 standard (refer to [ISO7816-3A1]). It is an asynchronous half duplex block transmission protocol. The main characteristics of the protocol are as follows:

- Commands may be initiated by both the IFD and the card.
- A block is the smallest data unit which can be transmitted between the card and the IFD.
- The correct reception of a block can be checked before processing the data delivered by the block.
- Recognizing the start and the end of a block is handled in the character component of the data link layer (to be discussed later).
- The protocol starts immediately after the ATR or the PTS with a block sent by the IFD and continues with alternating right to send a block.

According to the OSI reference model, three layers have been defined:

- **Physical layer** - Transfers bits with the same characteristics (bit rate, I/O states) as used during the ATR.
- **Data link layer** - Defines the character component and the block component. The character component exchanges characters as described for T=0, but without error detection, error signaling and character repetition. This implies a reduced delay between consecutive characters.
- **Application layer** - Processes commands, consisting of one or more blocks in each direction.

A block is a series of characters and contains the following fields: a prologue, an information (optional) and an epilogue field. The structure of a block is shown in Figure 3-7.
A brief description of the block fields is given below:

- **Prologue field:**
  - *Node address* - Is used to identify the source and the destination of the block. Bits b1 to b3 indicate the source node address (SAD) and bits b5 to b7 indicate the destination node address (DAD). Bits b4 and b8 are used for VPP state control (idle or active). A combination of a SAD and a DAD establishes a logical connection. More than one logical connection can exist at the same time.
  - *Protocol control byte* - Defines the type of the block. Three different types are specified:
    - Information block (I-block) - used to transfer information for use by the application layer and in addition an acknowledgment.
    - Receive ready block (R-block) - used to transfer acknowledgments, without using its information field.
    - Supervisory block (S-block) - used to exchange information between the chipcard and the IFD.
  - *Length* - Indicates the number of bytes transmitted in the information field.

- **Information field** - When present, this field contains application data (I-blocks) or control and status information (S-blocks).

- **Epilogue field** - This field is mandatory and contains an LRC (longitudinal redundancy check) or a CRC (cyclic redundancy check). The LRC is one byte, the CRC is two bytes in length.

The specific parameters in the ATR, as shown in Figure 3-6, are used to specify some T=1 related parameters. This involves the maximum supported information field size for the card and the IFD, waiting times between characters and blocks, and the error detection method. The coding of these specific parameters is beyond the scope of this report.

After the ATR or PTS, the IFD transmits either an I-block or an S-block to the chipcard. The chipcard answers by sending an acknowledgment. From that moment the IFD and the chipcard send blocks in an alternating way. Every reception of an I-, S- or R-block has to be acknowledged, before the next block can be sent.

<table>
<thead>
<tr>
<th>Prologue Field</th>
<th>Information Field</th>
<th>Epilogue Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node address (NAD)</td>
<td>Protocol control byte (PCB)</td>
<td>Length (LEN)</td>
</tr>
<tr>
<td>1 byte</td>
<td>1 byte</td>
<td>1 byte</td>
</tr>
</tbody>
</table>

1 or 2 bytes
I-blocks and R-blocks contain a sequence number consisting of one bit, that is used to implement data toggle synchronization. This mechanism guarantees data sequence synchronization between data transmitter and receiver across multiple transactions. At the start of a transaction the transmitter and receiver need to synchronize their sequence bits. The receiver toggles its sequence bit when it receives an error free block. The transmitter toggles its sequence bit only when it receives a valid acknowledgment. Figure 3-8 shows how this mechanism can resolve transaction errors.

The numbers in the circles represent the transmitter and the receiver sequence bits. During transfer i the transmitter sends data to the receiver. The receiver accepts the data and sends an acknowledge handshake, but the transmitter doesn’t receive the acknowledge as a result of a transmission error. That is why the transmitter doesn’t toggle its sequence bit and resends the previous data block. The receiver detects a data toggle failure and resends the acknowledge handshake, without toggling its sequence bit. This time the acknowledge is successfully received by the transmitter. Consequently, the transmitter toggles its sequence bit and sends the next data during transfer i+1. The reception is acknowledged by the receiver. The transaction has completed.

An S-block carries no sequence bit. Different S-blocks have been defined, to perform the following operations:

- **Resynchronize** - The synchronization numbers will be reset to their initial value.
- **Information field size adjustment** - Establishes a new maximum information field length.
- **Abort chain** - Aborts a chain (will be discussed subsequently).
- **Waiting time extension** - Establishes a new waiting time (time between two blocks).
- **VPP error response** - The IFD informs the chipcard of a VPP error.
Chaining is used when information, longer than the maximum supported information field size (IFS), has to be transmitted. The information is divided in parts with a size smaller or equal to IFS. The parts will be transmitted in separated blocks, and each block has to be acknowledged before the next block can be transmitted. To indicate that the next block is part of the chain, the M-bit (More data bit) in the PCB is set. In the last block the M-bit is reset to indicate it is the last block of the chain. As mentioned before, a chain can be aborted by sending an ‘abort chain S-block’.

For more information about T=1 refer to [ISO7816-3A1].

3.4.2 ISO 7816-4: Interindustry Commands for Chipcards

Part 4 of the ISO 7816 standard is called ‘Interindustry commands for interchange’. Except for commands transmitted between the chipcard and the IFD, the structure of data, access methods to data, security architecture and the content of the historical bytes is discussed.

3.4.2.1 Data Structures

Three types of files have been defined:

- **Master file (MF)** - The root file, is actually a mandatory dedicated file.
- **Dedicated file (DF)** - Contains file control information, and optionally memory available for allocation.
- **Elementary file (EF)** - Set of data units or records which share the same file identifier.

In Figure 3-9 an example of the logical file organization is shown.

![Figure 3-9: Logical file organization](image)
Two types of EFs have been defined:

- **Internal EF** - Intended for internal use. The card analyses and uses the data for management and control purposes.

- **Working EF** - Intended for external use. The data is not used by the card itself, but by the external world exclusively.

Four different methods are specified to select a file, namely:

- **Referencing by file identifier** - Any file can be referenced by using its unambiguous file identifier, consisting of two bytes. The MF’s identifier is ‘3F00’. The value ‘FFFF’ is reserved for future use. The value ‘3FFF’ is reserved for referencing by path.

- **Referencing by path** - Any file can be referenced by using a path to the file. A path is a concatenation of file identifiers, and begins with the identifier of the MF or the current DF. The value ‘3FFF’ can be used if the current DF is not known.

- **Referencing by short EF identifier** - Any EF may be referenced by using a short EF identifier in the range from 1 to 30 (coded on 5 bits). The currently selected EF can be referenced by using the value 0.

- **Referencing by DF name** - Any DF may be referenced by using a DF name coded on 1 to 16 bytes.

Four types of structures have been defined for EFs, and every chipcard should at least support one of them. The types of structures are:

- **Transparent structure** - The EF is seen as a sequence of data units.

- **Record structure** - The EF is seen as a sequence of individually identifiable records. Three different types of record structures have been defined:

  - Linear, fixed size records
  - Linear, variable size records
  - Cyclic, fixed size records

Figure 3-10 shows the four structures.
Data units can be referenced by an offset from the beginning of the EF. By default, the size of a data unit is one byte. Records can be referenced by a record identifier or by a record number.

3.4.2.2 Security Architecture

Security is an important aspect of the chipcard technology. Without sufficient security capabilities, there is no future for chipcards. In ISO 7816-4 the following security mechanisms have been defined:

- **Entity authentication with password** - The user needs to enter a password (i.e. a PIN code) before the card or an application can be used.

- **Entity authentication with key** - This is used to verify the outside world (e.g. card terminal or application). The card sends a random number to the external environment, which performs a cryptographic calculation using a certain key. The result is returned to the card, which checks the cryptogram.

- **Data authentication** - Both the outside world and the card add cryptographic redundancy to the data it transmits. The receiver checks this redundant information to verify if the received data is sent by an authorized entity.

- **Data encipherment** - Secret data is enciphered by the transmitter and deciphered by the receiver using a certain key. This reduces the possibilities of filtering messages by unauthorized parties.

3.4.2.3 Commands and Responses

ISO 7816-4 defines the structure of command messages and the resulting response messages. Both commands and responses are transmitted as Application Protocol Data Units (APDUs). The structure of a command APDU is shown in Figure 3-11.
The header is mandatory and consists of four bytes:

- **CLA** - Instruction class byte
- **INS** - Instruction code
- **P1** - Parameter byte
- **P2** - Parameter byte

The body is optional and it's length is variable. If the body contains a data field, then \( L_e \) indicates the number of bytes in the data field. If response data is expected, then \( L_e \) indicates the maximum number of bytes in the data field of the response APDU.

By default the \( L_c \) and \( L_e \) fields are only one byte in length. If \( L_c \) only contains zero's, then the two following bytes code \( L_e \). In this way the \( L_e \) field can be extended. \( L_e \) can also be extended up to 3 bytes. Because the receiver of the command APDU knows the length of \( L_c \) and of the data field, the length of \( L_e \) can always be determined.

The response APDU structure is shown in Figure 3-12

The maximum number of bytes in the data field are conform \( L_e \). The actual length can be less than \( L_e \) and is denoted by \( L_r \). The data field (i.e. the body) is optional. The trailer, consisting of two status bytes SW1 and SW2, is mandatory. The status bytes denote the processing state in the card. Figure 3-13 gives an indication of the coding of the status bytes. For detailed information about the coding of SW1 and SW2, refer to [ISO7816-4].
3.4.2.4 Logical Channels

Logical channels have been defined to establish logical links to DFs. Activity on one logical channel will be completely independent of activity on another logical channel. Commands refer to a certain logical channel by indicating the respective logical channel number in the CLA byte. ISO 7816-4 only supports activity on one logical channel at the same time. Only when the current command-response pair has been completed the subsequent command-response pair can be initiated.

A default logical channel is always available, and is called the basic logical channel. Additional logical channels can be opened and closed by using certain commands (to be discussed later).

3.4.2.5 Commands

This subsection gives a brief overview of the commands that have been defined in ISO 7816-4. Table 3-4, on the next page, shows the commands and describes their functions.

3.4.2.6 Historical Bytes

The historical bytes have been mentioned in section 3.4.1.3. However, the content of the historical bytes has not been discussed yet. Historical bytes can be used by the card to provide the IFD during the ATR with the following information:

- **Country code of the issuer**

- **Issuer identification number**

- **Card service data** - Describes the methods available on the card for supporting application-independent services (described in the next subsection).

- **Initial access data** - This data object allows the retrieval of a string of data objects (refer to the next subsection).

- **Card issuer's data** - This data object contains data with a structure and coding defined by the card issuer.

- **Pre-issuing data** - May be used to indicate the card manufacturer, IC type, IC manufacturer, ROM mask version and operating system version.

- **Card capabilities** - Informs the IFD of some capabilities of the card.

A detailed description of the historical bytes is beyond the scope of this project.
Table 3-4: Commands

<table>
<thead>
<tr>
<th>COMMAND</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>READ BINARY</td>
<td>Causes a response APDU containing (part of) the contents of an EF with transparent structure.</td>
</tr>
<tr>
<td>WRITE BINARY</td>
<td>Initiates the writing of binary values into an EF with transparent structure. The contents already present in the EF can be AND-ed, OR-ed or replaced by the value in the command APDU using a one-time write.</td>
</tr>
<tr>
<td>UPDATE BINARY</td>
<td>Initiates the update of the data already present in an EF with the data in the command APDU.</td>
</tr>
<tr>
<td>ERASE BINARY</td>
<td>Sets (part of) the content of an EF to its logical erased state, starting from a given offset.</td>
</tr>
<tr>
<td>READ RECORD(S)</td>
<td>Gives the contents of the specified record(s) of an EF.</td>
</tr>
<tr>
<td>WRITE RECORD</td>
<td>Initiates the writing of records into an EF with record structure. The contents already present in the EF can be AND-ed, OR-ed or replaced by the value in the command APDU using a one-time write.</td>
</tr>
<tr>
<td>APPEND RECORD</td>
<td>Initiates the appending of a record to an EF and sets the record pointer to the successfully appended record.</td>
</tr>
<tr>
<td>UPDATE RECORD</td>
<td>Initiates the update of the record already present in an EF with the data in the command APDU.</td>
</tr>
<tr>
<td>GET DATA</td>
<td>Used to retrieve one primitive data object, or one or more data objects contained in a constructed data object.</td>
</tr>
<tr>
<td>PUT DATA</td>
<td>Used to store one primitive data object, or one or more data objects contained in a constructed data object.</td>
</tr>
<tr>
<td>SELECT FILE</td>
<td>Sets a current file within a logical channel. Subsequent commands may refer to that file by indicating the logical channel number.</td>
</tr>
<tr>
<td>VERIFY</td>
<td>Initiates the comparison of data received from the interface device by the card with the reference data stored in the card (e.g. password).</td>
</tr>
<tr>
<td>INTERNAL AUTHENTICATE</td>
<td>Initiates the computation of the authentication data by the card using the challenge data sent by the interface device and a secret stored in the card (e.g. a key).</td>
</tr>
<tr>
<td>EXTERNAL AUTHENTICATE</td>
<td>Updates the security status using the result of a previously challenge sent by the card to the interface device (i.e. the authentication data transmitted by the interface device) and a secret stored in the card (e.g. a key).</td>
</tr>
<tr>
<td>GET CHALLENGE</td>
<td>Issues a challenge (e.g. random number) for use in a security related procedure.</td>
</tr>
<tr>
<td>MANAGE CHANNEL</td>
<td>Opens and closes logical channels other than the basic logical channel.</td>
</tr>
<tr>
<td>GET RESPONSE</td>
<td>Is used to transmit (part of) APDUs from the card to the interface device, which otherwise could not be transmitted by the available protocols.</td>
</tr>
<tr>
<td>ENVELOPE</td>
<td>Is used to transmit (part of) APDUs, or any data string, which otherwise could not be transmitted by the available protocols.</td>
</tr>
</tbody>
</table>
3.4.2.7 Application-Independent Card Services

A set of application-independent services has been defined. These services provide mechanisms between a card and an IFD, which know nothing from each other. The following application-independent services have been defined:

- **Card identification service** - Allows the IFD to identify the card and determine how to deal with it. The card identification data is provided by the card by means of the historical bytes or a file implicitly selected immediately after the ATR.

- **Application selection service** - Allows the IFD to determine what application is active in the card (if any), and to select and start an application in the card.

- **Data object retrieval service** - Allows to retrieve data objects.

- **File selection service** - Allows selection of unnamed DFs and EFs.

- **File I/O service** - Allows access to data stored in EFs

The following paragraph provides a brief review of the ISO 7816 standard.

3.5 Summary and Conclusions

In this chapter, the reader has been given a lot of information about chipcards, like a typical architecture, security methods like DES, RSA and digital envelope. The most widely used chipcard standard, ISO 7816, has been described as well.

To conclude this chapter, we will briefly review the ISO 7816 standard. In the context of this report, part 3 and 4 are most important, and have been discussed extensively. In section 3.4.1 we took a look at the electrical interface, the Answer To Reset and the two defined transmission protocols T=0 and T=1. In section 3.4.2 the main subjects were the data structures defined for chipcards, security mechanisms and the specified commands and responses.

The ISO 7816 standard forms a good basis for the chipcard technology. It gives the reader a good idea of the techniques used in chipcards and terminals. However, the standard is not very detailed and often allows an own interpretation. Sometimes, the standard explicitly mentions that cards do not have to comply to certain parts of the standard. This introduces an enormous flexibility for card developers, but it reduces compatibility among chipcards. In practice, most chipcards will comply to the standard, but the different cards are not likely to be compatible. In order to achieve global compatibility among chipcards, another, more detailed standard, is required. And this standard should be accepted worldwide. In addition to the ISO 7816 standard, two other important standards exist, namely the EMV standard and the ETSI TE9 standard. Both standards are gaining popularity, but cause a separation in chipcard technology. At least both standards comply to the ISO 7816 standard, which again emphasizes its importance.
4. Charging

Of course customers have to pay for provided IN/CC services. Different payment methods can be used, as will be discussed in this chapter. The purpose of the provided information is to give the reader an overview of the payment methods. An extensive discussion of the methods is out of the scope of this report.

4.1 Introduction

Customers have to be charged for using IN/CC services. The used charging method depends on the payment method(s) that the card supports and which payment method the customer prefers. If the card supports more than one method, then the user has to select one of the supported methods. The selected method must be known in both the card reader and the network host, which manages the charging (at least for IN/CC services). The host determines the amount of money that the user owes the service provider. This amount is dependent on the used type of service and the used quantity of service (e.g. time-proportional). How the costs for using a service are determined is out of the scope of this report. However, we will take a look at the different payment methods.

4.2 Post-payment

When applying the post-payment method, the user is charged after having used a service. The owed amount of money is debited from the customer’s banking account. Two different post-payment methods exist:

- **Terminating point dependent** - The charged banking account is dependent on the terminating point from which the customer initiated an IN/CC service. So whoever used the service, the owner of the terminal equipment is charged.

- **Card dependent** - The charged banking account is dependent on the used chipcard. The chipcard contains information about the banking account that should be charged. So the owner of the chipcard is charged, no matter from which terminal equipment the service was used.

The first method is the classic way of charging customers for using telephone services. The second way can only be used when the customer has a chipcard on which certain banking information is stored. This method is for instance used in the Account Card Calling (ACC) IN/CC service, which will be discussed in 5.4.1.1.

A typical file structure of the ACC application on the card is depicted in Figure 4-1.
The MF contains an EF_{DIR} which contains a list of supported applications. One of these applications is ACC and all related data is stored in DF_{ACC}. The DF_{ACC} typically contains the following three EFs:

- **EF_{PIN}** - This optional EF contains the Personal Identification Number (PIN). If EF_{PIN} is present in the DF, then the user must authenticate himself by entering his PIN. The entered PIN is compared to the PIN in EF_{PIN}. The user can only access the application if the entered PIN is correct.

- **EF_{KEY}** - This EF contains the keys, which are used for authentication of the card and the network host. The corresponding keys must be present in the host as well.

- **EF_{PST}** - This post-payment EF contains all information that is necessary to use the ACC service. This information at least contains a banking account number. Additional information is, for instance, the name of the bank or the expire date.

The information in the EF_{PST} can only be accessed after the user is authenticated by entering the proper PIN (if required) and after both the card and the host have been authenticated by using the keys. This may be done in a DES or RSA authentication process.

By accessing the EF_{PST} the host acquires the information it needs to apply the post-payment method. Optionally, the host may verify the banking account number to ensure it matches the card ID, the banking account is not blocked, and the card has not been involved in fraudulent transactions. If everything seems to be all right, the service can be provided. The costs of the service will be billed to the proper banking account.
4.3 Pre-payment

When applying the pre-payment method, the user has to pay for a service before he actually uses it. This is done by converting paid money to equivalent electronic money on a chipcard. The amount of money owed for using a service, is deducted from the electronic purse. To illustrate the ‘circle of money’ for the pre-payment method, Figure 4-2 is depicted.

As shown, money is withdrawn from a banking account to charge the chipcard’s electronic purse. When a customer pays for a service with his chipcard, electronic money flows from the card to the terminal. After a while (e.g. once every week) all electronic money, collected in the terminal, will be transferred to the network host. The host collects all electronic money from all terminals connected to the network, and deposits it on the banking accounts of the respective service providers.

Two different pre-payment methods exist:

- **Single transaction** - The costs of a service are debited from the electronic purse in one single transaction. This can be done before the service is used, on condition that the costs of the service are known in advance. Otherwise, the electronic purse is debited afterwards.

- **Sliced transaction** - The costs of a service are debited from the electronic purse slice by slice. This means that after a certain quantity of service has been provided (e.g. after a certain time), the electronic purse is debited with an equivalent amount of money. The sliced payment can also be applied before the equivalent service quantity has been provided.

Both pre-payment methods are safer, if the payment happens before the service is provided. This eliminates the risk of not receiving any payment after the service has been provided. This risk is relevant, because the user could remove the card from the card reader before the electronic purse had been debited, or payment might be impossible due to an error.
Another issue regarding the sliced pre-payment is the size of the slices. The network has to indicate when the next slice should be debited from the electronic purse. For instance, this can be done after a certain amount of bandwidth has been used or after a certain amount of time. The network may use some kind of in-band signaling (e.g., Data Over Voice) to initiate the next slice transaction. Another possibility is that the host informs the terminal about the size of the slices in advance. In this situation the terminal itself initiates the next slice transaction. At that moment the card reader starts a conversation with the chipcard. After a while the electronic purse has been debited and the card reader is ready for the next slice.

It is clear that the time between every slice should not be too short, because then the card reader cannot catch up with the network. A disadvantage of large inter-slice-intervals is that there is a big change that the customer pays more than he uses. In other words, the last paid slice has not been used completely. If this is the case, some facilities are necessary to overcome this problem. The remainder of the slice may be paid back to the customer.

Figure 4-3 illustrates what a typical pre-payment file structure looks like.

![Pre-payment file structure](image)

The DFPRE typically contains four EFs:

- **EFPIN** - This EF contains the PIN, which is used to authenticate the user. He can only get access to the pre-payment application by entering the correct PIN.

- **EFKEY** - This EF contains the keys, which are used to authenticate both the card and the terminal. In the terminal the corresponding keys must be present.

- **EFPRE** - The EFPRE can contain information like expire date, pre-payment serial number, and unit equivalence. The pre-payment serial number can be verified, to determine if the number is in the red list. The unit equivalence indicates what the value is of one unit in the electronic purse.

- **EFNOU** - This EF indicates the amount of electronic money stored on the card. More accurate, it contains the Number Of Units (NOU) stored on the card. The equivalent amount of money can be determined by multiplying the unit equivalence with the number of units.
The EF$_{PN}$ and the EF$_{KEY}$ are introduced for security purposes. The EF$_{PRE}$ and the EF$_{NOU}$ contain the necessary information to apply the pre-payment method. When the user pays for a service, the number of units in EF$_{NOU}$ is decreased. When the user loads his card, the number of units is increased.

In Table 4-1 is shown, what a typical information flow for the pre-payment method looks like.

<table>
<thead>
<tr>
<th>Step</th>
<th>SAM</th>
<th>Terminal</th>
<th>Chipcard</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Answer To Reset</td>
<td>Reset</td>
<td>Answer To Reset</td>
</tr>
<tr>
<td>2</td>
<td>Return random</td>
<td>Ask random, Authenticate card</td>
<td>Return AC</td>
</tr>
<tr>
<td></td>
<td>Verify AC</td>
<td>Verify AC</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Return AC</td>
<td>Ask random, Authenticate SAM</td>
<td>Return random</td>
</tr>
<tr>
<td></td>
<td>Verify AC</td>
<td>Verify AC</td>
<td>Verify AC, Return result</td>
</tr>
<tr>
<td>4</td>
<td>Return random</td>
<td>Ask random, Single debit</td>
<td>Single debit, Generate debit AC, Return result</td>
</tr>
<tr>
<td></td>
<td>Verify debit AC</td>
<td>Credit SAM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Credit SAM</td>
<td>Single debit OK</td>
<td></td>
</tr>
</tbody>
</table>

In the first step, both the chipcard and the SAM are reset and an Answer To Reset (ATR) is returned to the terminal. After this step has completed successfully, the terminal can communicate with the SAM and the chipcard. During the second step, the card is authenticated by the SAM and subsequently the SAM is authenticated by the card. During the final step, the actual debit is performed. The card calculates an Authentication Code (AC) by using the random number from the SAM and the debited amount. The SAM verifies this AC to make sure that the card has been debited with the proper amount. Subsequently, the SAM is credited with the same amount, and the transaction has completed.

The shown information flow applies for the single transaction. However, the sliced transaction is very similar. The main difference is that step four will be repeated for every slice.
The discussed information flow does not provide a detailed description of the pre-payment method, but it gives a good idea of how a typical transaction works. The implementation of the pre-paid method in the Chipknip and the Chipper has been studied, but will not be described in this document, since the information is classified. However, this information has been described in an separate classified document ([HELV97]), which will only be distributed to those who are entitled to get this information.

4.4 Summary and Conclusions

In this chapter four different payment methods have been considered. These four methods can be divided into two categories: post-payment and pre-payment. In case of post-payment, the user is charged after having used a service. The charging may be terminating point dependent or card dependent. In pre-payment transactions, a chipcard is always involved. The user loads his card with money, which is withdrawn during a pre-payment transaction. This can be done in one transaction (i.e. single transaction) or in different subtransactions (i.e. sliced transaction).
5. IN/CC Services

After having discussed the IN services specified in the Q.12xx series in chapter 2 and the capabilities of chipcards in the previous chapters, we will now take a look at IN services that can be implemented by using chipcards. These IN/CC services will be derived from the specified IN services. First of all, selection criteria will be formulated, and these will be used to select useful IN services. The selected IN services form the basis for the set of IN/CC services. Finally, these IN/CC services will be described.

5.1 Introduction

As mentioned in chapter 1, the objective of this project is to investigate the possibilities of using chipcards in IN services. The basis for this research is the identification of IN services that can make use of chipcards. These IN/CC services are the topic of this chapter.

Before the intended IN services can be selected, some selection criteria should be formulated. Important in this context is, what the added value of chipcards is in IN services. As described in the previous chapter, chipcards are useful when security is important. Cryptographic calculations can be performed to transmit encrypted data and to authenticate the source and destination of the information flow. Further, a lot of information can be stored on a chipcard. It might be convenient for the user of IN services to provide IN with the necessary information by means of a chipcard instead of by using a telephone. A chipcard can contain alphanumeric information, which is almost impossible to generate by means of a telephone. And the information stored on a chipcard can be generated and changed by the user by using a PC. Compared to a voice responding system, the PC not only provides a user friendly interface. The editing can be done off-line as well, and that reduces telephone costs for the user. Finally, the card could be used to upload the service profiles to different networks from different network operators. So if a user wants to use the services from more than one national operator or when he is traveling to other countries, he can still use his own service profiles.

In the next paragraph the selection criteria will be formulated.

5.2 Selection Criteria

This paragraph discusses the selection criteria that will be used to select IN services, which can be extended to IN/CC services. As will be illustrated, the criteria’s most important aspects are information (I), security (S), user interaction (U) and efficiency (E). The criteria are referenced by I, S, U and E criteria respectively. The I criteria determine if information needs to be provided by the user. The S criteria determine whether or not security is necessary. The U criteria determine whether or not the information has to be provided and/or changed by the user frequently. Finally, the E criterion determines if the use of a chipcard, to provide the information, is more efficient than providing the information without a chipcard.
The intended IN services should at least meet one of the following I criteria:

**Information**

- **I.1** - To deliver an IN service successfully, some information *about* the user is necessary.
- **I.2** - To deliver an IN service successfully, some information *from* the user is necessary.

Information *about* the user is personal information like a banking account number or Personal Telecommunication Number (PTN). Information *from* the user is information which can be determined and changed by the user like a forwarding number or a list of phone numbers that are not allowed to be dialed.

If one or both of the I criteria are met, the following S and U criteria must be examined:

**Security**

- **S.1** - The transfer of information should be secure.
- **S.2** - The source of the information should be authenticated.
- **S.3** - The destination of the information should be authenticated.

As appeared in the previous chapters, chipcards can perform complex cryptographic calculations, which might be useful to make certain IN services more secure. Not only the transmitted data can be coded, also the source and destination of the information flow can be verified. If at least one I criteria is met and subsequently one or more S criteria are met, the examined IN service may be selected.

**User Interaction**

- **U.1** - The same information has to be provided by the user frequently.
- **U.2** - The information may be changed by the user occasionally.

If the user has to provide the same information or change the information often, chipcards might be convenient. In the first case, the fixed information can be stored on the card and be transmitted when requested by IN. In the latter case, the variable information can be stored on the card. If the user wants to change the information, he can load, edit and store the data. In this way, the information can be reused, and re-editing all data is not necessary. Whether or not the service is selected when one or more U criteria are met, depends on a final criterion:

**Efficiency**

- **E.1** - Providing the information by means of a chipcard will be more efficient than providing the information by means of a telephone.

If the information has to be provided by means of a chipcard, the information must be generated or edited by using a PC. The user interface might be very convenient, but the editing process may be time consuming. The PC must be booted and software must be loaded. If this is more efficient than generating the information with a telephone, the service may be selected. This will often be the case, when a great amount of, especially alphanumeric, data has to be provided, and when the data can be reused.
To illustrate the use of the criteria, Figure 5-1 is depicted.

![Figure 5-1: Selection process](image)

In the following paragraph the criteria will be used to determine which IN services can be extended to IN/CC services.

### 5.3 Selection Process

The discussed selection criteria will be used to select IN services in this paragraph. Table 5-1 and Table 5-2 show all IN services as defined in CS-1 and CS-2 respectively. Illustrated is whether or not the selection criteria apply to the services and which services have been selected.

#### Table 5-1: Selection of CS-1 IN services

<table>
<thead>
<tr>
<th>CS-1 Service</th>
<th>Abbr.</th>
<th>L.1</th>
<th>L.2</th>
<th>S.1</th>
<th>S.2</th>
<th>S.3</th>
<th>U.1</th>
<th>U.2</th>
<th>E.1</th>
<th>Select</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abbreviated Dialing</td>
<td>ABD</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Account Card Calling</td>
<td>ACC</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Automatic Alternative Billing</td>
<td>AAB</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Call Distribution</td>
<td>CD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Call Forwarding</td>
<td>CF</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Call Rerouting Distribution</td>
<td>CRD</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Completion of Calls to Busy Subscriber</td>
<td>CCBS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conference Calling</td>
<td>CON</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Credit Card Calling</td>
<td>CCC</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Destination Call Routing</td>
<td>DCR</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Follow-Me Diversion</td>
<td>FMD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. The AAB service has not been selected, because the respective IN/CC service would be equal to the ACC service. The main difference between the ACC and the AAB service is that the AAB makes no use of a card.
### Chipcards in Intelligent Network Services

#### Table 5-2: Selection of CS-2 IN services

<table>
<thead>
<tr>
<th>CS-2 Service</th>
<th>Abbr.</th>
<th>L.1</th>
<th>L.2</th>
<th>S.1</th>
<th>S.2</th>
<th>S.3</th>
<th>U.1</th>
<th>U.2</th>
<th>E.1</th>
<th>Select</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call Hold</td>
<td>CH</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Call Transfer</td>
<td>TRA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Call Waiting</td>
<td>CW</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Future Public Land Mobile</td>
<td>FPL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Telecommunication System</td>
<td>MTS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Global Virtual Network Service</td>
<td>GVNS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hot Line</td>
<td>HOT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>International Telecommunication Charge Card</td>
<td>ITCC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internetwork Freephone</td>
<td>IFPH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internetwork Mass Calling</td>
<td>IMAS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internetwork Premium Rate</td>
<td>IPRM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internetwork Televoting</td>
<td>IVOT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Store and Forward</td>
<td>MSF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multimedia</td>
<td>MMD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Terminating Key Code Screening</td>
<td>TKCS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

### Chipcards in Intelligent Network Services

#### Table 5-2: Selection of CS-2 IN services

<table>
<thead>
<tr>
<th>CS-1 Service</th>
<th>Abbr.</th>
<th>L.1</th>
<th>L.2</th>
<th>S.1</th>
<th>S.2</th>
<th>S.3</th>
<th>U.1</th>
<th>U.2</th>
<th>E.1</th>
<th>Select</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freephone</td>
<td>FPH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Malicious Call Identification</td>
<td>MCI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mass Calling</td>
<td>MAS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Originating Call Screening</td>
<td>OCS</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Premium Rate</td>
<td>PRM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Security Screening</td>
<td>SEC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selective Call Forwarding on busy/don't answer</td>
<td>SCF-BY/DA</td>
<td>x</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selective Call Forwarding</td>
<td>SCF</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Call Forwarding on Busy</td>
<td>CFBY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Call Forwarding on Don't Answer</td>
<td>CFDA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Split Charging</td>
<td>SPL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Televoting</td>
<td>VOT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Terminating Call Screening</td>
<td>TCS</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Universal Access Number</td>
<td>UAN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Universal Personal Telecom</td>
<td>UPT</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User-Defined Routing</td>
<td>UDR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Virtual Private Network</td>
<td>VPN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**ERICSSON**

48
5.4 Description of selected IN/CC Services

In the previous paragraph, fifteen IN services have been selected. These IN services can be extended to IN/CC services. We will now take a closer look at these services. Two reasons for using chipcards in the selected IN services can be identified. Firstly, chipcards can be used to make a service more secure. Secondly, a service profile can be stored on chipcards. We now will divide the selected services in the two corresponding categories. For both categories, one service, ACC and CD respectively, will be discussed extensively. The other services will be described only briefly.

5.4.1 Security Services

To clarify how chipcards can improve the security of IN services, the ACC service is discussed in the following subsection. The subsequent subsections provide a short description of other IN services that can be made more secure with chipcards.

5.4.1.1 Account Card Calling

The Account Card Calling (ACC) service allows a user to call from any telephone to any destination number and have the costs for the call charged to his personal account. This service can be used without a card reading device. The user dials the access code and enters his card number and PIN. After this information has been verified and approved, the user can dial the destination number. No chipcard is needed to use the service.

However, this service can be used with a chipcard. The user inserts his chipcard in the terminal and dials the access code. The card number is provided by the card and the user only has to enter his PIN. The PIN is validated by the card, and if correct, authenticates the user. Subsequently, the card will verify the host in the network to which it is connected. This host contains a SAM. The chipcard and the SAM will authenticate each other to make sure both entities are real. The transferred data may be coded, to prevent a third party from filtering the telephone line. If all entities trust each other, the user can dial the destination number.

The advantage of using a chipcard, is the improved security of the service. First of all, the entered PIN is not transmitted over the network, and cannot be intercepted by a third party. So the authentication of the user is an off-line process. Secondly, all other secret information (e.g. card number and banking account number) can be encrypted before transmission. And finally, the authenticity of the user and both the chipcard and the host is guaranteed. This prevents fraudulent use of fake cards and the user is assured that his personal information is sent to a trusted party.
5.4.1.2 Credit Card Calling

The Credit Card Calling (CCC) service is almost equal to the ACC service. The only difference is, that ACC uses as special ACC card, and CCC can be used with a general purpose credit card. The customer has to provide his credit card number and a PIN, and the costs for the call are charged to the proper banking account.

By storing the credit card number on a chipcard, this number can be encrypted before transfer. Further, both the card and the host can be authenticated. Chipcards can make this service more secure.

5.4.1.3 Universal Personal Telecommunications

The Universal Personal Telecommunications (UPT) service provides the user with mobility, since the user has a unique Personal Telecommunication Number (PTN) which will be translated by the network to an appropriate destination number. The user has to notify IN of his location. From that moment, all calls to his PTN will be routed to this location.

By using a chipcard, the identification of the user and the authentication of the user, card and host will be secure.

5.4.1.4 International Telecommunication Charge Card

The International Telecommunication Charge Card (ITCC) service allows subscribers to make use of telecommunication services in the ‘visited network’ and being charged to the customer’s account number by the card issuer in the ‘home network’.

To secure this service, chipcards can be used.

5.4.2 Profile Services

A service profile may be stored on a chipcard to enable a user of IN services to easily store, change and provide certain information that is needed by the network to provide an IN service. To illustrate the use of a service profile, the CD service is described in the next subsection. The subsequent subsections give a brief description of the other IN services, which may use a service profile.

5.4.2.1 Call Distribution

The Call Distribution (CD) service can be used by the subscriber to have incoming calls routed to different destinations. The allocation of the calls is done according to three types of distribution algorithms:
Chipcards in Intelligent Network Services

- **Circular distribution** - The calls are routed to the different locations alternately. This results in a uniform load.

- **Percentage distribution** - The calls are routed to the different locations according to a percentage. For every destination is specified what percentage of the calls are routed to it.

- **Hierarchical distribution** - The calls are routed to the different locations using a priority list. The location with the highest priority gets the calls if the line is not busy.

Congestion at one location may cause calls to be routed to an alternative location.

The subscriber can manage the allocation process. Firstly, the telephone numbers of the different locations have to be specified. Subsequently, a distribution algorithm must be chosen. And finally, if the percentage distribution is used, the respective percentages of the locations should be specified. If the hierarchical distribution is used, a priority list must be provided.

A chipcard may be useful. The complete CD profile can be edited on a PC and stored on the chipcard. To activate the service, the profile must be uploaded to the IN. If the user wants to change the profile, he can load it from the card into his PC, edit the profile, and store it on the card again. Whenever he wants, he can upload the profile to any network’s central database.

5.4.2.2 Abbreviated Dialing

The Abbreviated Dialing (ABD) service enables users to use shorter telephone numbers. The user has to provide telephone numbers and specify their abbreviations. The lists with numbers can be stored on a chipcard.

5.4.2.3 Call Rerouting Distribution

The Call Rerouting Distribution (CRD) service allows subscribers to have incoming calls encountering a triggering condition (busy, specified number of rings, queue overload or call limiter) rerouted according to a predefined choice. The calls can be rerouted to another number, on an announcement, or queued.

A CRD profile, describing the triggering condition(s) and the rerouting method(s), can be stored on a chipcard.
5.4.2.4 Destination Call Routing

The Destination Call Routing (DCR) service enables subscribers to specify the routing of their calls to destinations according to:

- Time of day, day of week, etc.
- Area of call origination
- Calling line identity of customer
- Service attributes held against the customer
- Priority (e.g. from input of a PIN)
- Charge rates applicable for the destinations
- Proportional routing of traffic

The DCR profile indicates which of the above mentioned conditions are used and specifies condition specific parameters.

5.4.2.5 Originating Call Screening

The Originating Call Screening (OCS) allows users to specify a list of telephone numbers that are or are not allowed to be called. Optionally, the time of day for which this list applies may be indicated. Further, the control can be overridden on a per-call basis by anyone with the proper PIN.

The telephone list, the time of day for which it applies and the PIN form the OCS profile.

5.4.2.6 Selective Call Forwarding on Busy/Don’t Answer

The Selective Call Forwarding on Busy/Don’t Answer (SCF-BY/DA) service allows pre-selected calls to be forwarded if the called user is busy or does not answer the phone within Y seconds or X rings. The SCF-BY/DA list contains numbers from callers and the respective forwarding numbers. For callers not in the list, a default forwarding number is specified. Also a time of day condition can be applied.

The SCF-BY/DA profile contains the SCF-BY/DA list and the time of day conditions.

5.4.2.7 Selective Call Forwarding

The Selective Call Forwarding (SCF) service, is very similar to the SCF-BY/DA service. The difference is that SCF forwards pre-selected calls, no matter what the status of the called party line is. Further, time of date conditions cannot be applied.

The SCF profile contains the SCF list with pre-selected numbers of callers and the respective forwarding numbers.
5.4.2.8 Terminating Call Screening

The Terminating Call Screening (TCS) service, resembles the OCS service. Instead of originating calls, terminating calls are screened. Incoming calls may be restricted or allowed, according to the screening list and optionally, by the time of day.

The TCS profile contains the screening list and the time of day condition parameters.

5.4.2.9 Universal Access Number

The Universal Access Number (UAN) service enables a subscriber with several terminating lines in different locations, to have one unique number. Incoming calls to the unique number are routed to the terminating lines based upon the area of originating call.

The link between area codes of originating calls and the called terminating lines can be specified in a UAN profile.

5.4.2.10 Hot Line

The Hot Line (HOT) service allows a user to call a certain destination without entering the telephone number. This number is stored in the network by prior subscription. Further, incoming calls are screened and only accepted when contained in a screening list.

The HOT profile contains the screening list.

5.4.2.11 Message Store and Forward

The Message Store and Forward (MSF) service allows users to send messages (e.g. voice, data, fax) to one or more destinations. Different methods of delivery and/or times of delivery may be specified.

The MSF profile contains the methods of delivery and the times of delivery.
5.5 Summary and Conclusions

Several IN services, as specified in CS-1 and CS-2, can be extended to IN/CC services. These services make use of chipcards to make the corresponding IN services more secure or to enable easy editing and portability of service profiles. To determine which IN services qualify for extension to IN/CC services, four groups of selection criteria have been identified, namely information (I), security (S), user interaction (U) and efficiency (E) criteria. By means of these criteria, fifteen services have been selected.

The selected services can be divided into two groups. The first group contains services that can use chipcards to make the service more secure. Personal, secret information is always involved in these services. The second group contains services that can use chipcards to store and change service profiles. The content of these service profiles is determined and managed by the user.

As appeared in this chapter, chipcards can provide more security and ease of use. Introducing chipcards in IN services might encourage customers to utilize the services, both within and outside their operators network.
6. IN/CC Environment

In the previous chapters both the IN and the chipcard technology have been discussed. Also possible IN/CC services have been selected. In this chapter the IN/CC environment will be discussed. A general IN/CC architecture and an architecture used to prototype IN/CC services will be described. Subsequently, the communication protocols for these architectures, INAP respectively TCP/IP, are discussed. This chapter is concluded with a brief description of the Erlang messages, that have been defined to be transferred between the different entities of the IN/CC environment.

6.1 Introduction

In this chapter the IN/CC environment will be described. The architecture in a general IN environment and in the prototyping environment in the IN Application Laboratory (INAL) will be considered. The ultimate goal is to have a chipcard communicate with the IN. At the card side, at least a chipcard terminal should be present. Information will flow from the card to the network, via the terminal, or in the opposite direction. The card can communicate with the SRF in the network. The SRF gets its information from or distributes the received information to the SCF and the SDF.

In the following sections, we will pay attention to the involved network entities, the information flows, and the used protocols.

6.2 General Architecture

Before we focus on the prototyping architecture, we will consider a more general situation. Figure 6-1 depicts an architectural overview of a general IN/CC environment.

![Figure 6-1: General architectural overview](image)

At the user's side, at least a terminal must be present. Optionally, a PC can be used to control the terminal's behavior and to control a modem or an ISDN card. When no PC is used, the terminal should be 'intelligent' and contain some software and a modem or ISDN interface.
The other elements in the picture have already been described briefly in chapter 2. Both the CCAF and the CCFs handle non-IN calls. The SSF, SCF, SRF and SDF will only be involved in IN calls. The SSF recognizes IN calls and gives the SCF control over the call. The SRF is used for all user interaction. Finally, the SDF is a database which contains, among other things, authentication data, user profiles and service profiles.

The SRF is part of the Intelligent Peripheral (IP). The IP contains a CCF, a SAM and optionally a modem. A modem is needed to enable communication with a possible modem at the user's side. Digital data from the chipcard is converted to analog signals by the modem at the user's side. These analog signals can be transmitted via a normal speech connection to the IP. The modem in the IP converts the analog signals back to digital data, which the SRF can deal with. If ISDN is used, of course no modems are needed. The communication process then is completely digital.

The SAM contains secret keys corresponding to the keys in the chipcard. It has the capability to perform cryptographic calculations. The SAM is used to authenticate both the SRF to the card and the card to the SRF. Further, the SAM can encode and decode data that should be transmitted encrypted.

In brief, when the user dials a certain service access number (e.g. 0900 number), the SSF recognizes the number and allows the SCF to take over the control of the call. When user interaction is required for the respective service, a service assist procedure is started by the SCF. This means that a connection to the SRF in the IP is established. From this point, the card and the SRF can communicate.

The service assist procedure is a feature from the USER INTERACTION SIB. Usually, this SIB is used to enable user interaction. For instance, it can be used to play announcements or to collect information from the user. Information collection mostly is performed by sending DTMF tones to the SRF. However, this feature is limited to transferring numbers and is not useful in IN/CC services, since it should be possible to transfer complex authentication codes and characters. So for IN/CC services, the SIB is only useful for its service assist procedure. To transfer the required information, a new SIB is needed.

Data is transferred between the SDF and the chipcard via the SRF and the SCF. The SCF uses the SERVICE DATA MANAGEMENT SIB to communicate with the SDF. This SIB provides the capability for the SCF to search and retrieve data and modify data in a specified storage space in the SDF.

To give the reader an idea of the information flow between the different entities, Figure 6-2 is depicted. This figure shows how a connection between the SRF and SSF is established. Subsequently, a script is started on the SRF for authentication purposes and to find out whether a download or an upload session is required. The figure also shows a download and an upload session.
The bold operations in the figure represent DSS1 operations. DSS1 is an abbreviation for Digital Subscriber Signaling System No. 1. DSS1 is the protocol used on the network layer of an ISDN D-channel.

The first operation is the dialing of the service number. Note that this number may be dialed by a card application or by the user himself.
Until the Run Script instruction is sent from the SCF to the SRF, the information flow is reasonably standard. The SCF gets control over the call and initiates the service assist procedure, to establish a connection with the SRF. The Run Script instruction is not standardized. It hands the control over to the SRF, which starts a conversation with the card. All involved parties are authenticated and the card sends a user ID number to the SRF, which is used to identify the user. The required transfer direction of profiles (upload or download) is indicated as well. The SCF searches for the user ID in the SDF and gets access to the relevant profiles. Subsequently, the profiles are uploaded or downloaded. From that point, the SRF will not be involved anymore, and the connection with the SRF is released.

6.3 Prototyping Architecture

Figure 6-3 depicts the prototyping architecture of the IN/CC environment.

At the user's side, the figure shows a terminal, a PC, a modem and a telephone. Communication between the card and the IN is done via an analog connection by means of two modems. One modem is connected to the PC, the other modem is connected to the IN prototyping environment in the INAL, also called the Rapid Service Prototyping Environment (RSPE). The RSPE consists of the SSP, IP, SCF and SDF. A SMF and SCEF are also part of the RSPE, but these entities have not been depicted in Figure 6-3. The RSPE completely implements IN in software. The IN entities can be executed on different SUN workstations, or all entities can be executed on a single workstation.

Different from the general architecture is the way the IN entities communicate. In the general situation the SCF communicates with the SDF, SRF and SSF via the INAP protocol, while in the RSPE, data is transferred by sending Erlang messages. If the entities are executed on different workstations, communication is performed by sending Erlang messages via TCP/IP.
Erlang\(^1\) is a programming language developed at the Ericsson and Ellemtel Computer Science Laboratories. It is especially suited for programming fault-tolerant real-time applications.

The INAP and TCP/IP protocols will be discussed in the following paragraph.

## 6.4 Communication Protocols

In the previous paragraph, two communication protocols have been mentioned, namely INAP and TCP/IP. We will briefly discuss these protocols in this paragraph.

### 6.4.1 IN Application Protocol

The IN Application Protocol (INAP) has been developed to make communication between the SCF and SSF, SDF and SRF possible. For CS-1 INAP is specified in [Q.1218], and for CS-2 it is specified in [Q.1228].

As shown in Figure 6-4, INAP is an application layer protocol of the SS7 protocol.

![Figure 6-4: SS7 protocol architecture](image)

The MTP, shown in Figure 6-4, handles the functionality of the OSI physical, data link and partly the network layer. It provides reliable transfer of signaling messages to the higher layers. One of these layers is the TUP. It has been defined for the control of telephone calls, but since it is very limited, the ISUP has been introduced. The DUP has been developed for data communication signaling.

\(^1\) For more information about Erlang refer to internet: http://erlang.ericsson.se
In the IN concept not only signaling for end-to-end communication is needed, but signaling for communication between network elements is necessary as well. The SCCP implements explicit addressing of these network entities and more sophisticated services between them.

In the OSI application layer, the INAP is located on the TCAP. TCAP/INAP provides an open standard, and enables different manufacturers and network operators to develop equipment and services that are compatible.

For the SCF-SSF, SCF-SDF and SCF-SRF communication, operations have been defined using the Abstract Syntax Notation 1 (ASN.1). A typical ASN.1 representation of an INAP operation is shown in Figure 6-5.

The bold words in the figure are ASN.1 keywords. The other words represent variables which actually describe the INAP operations. The left-hand block is the main part of the description. It specifies the operation name, and a link to a list of arguments and a list of results. Further, links can be specified to other operations, and finally a list of possible errors can be specified. The middle block shows a possible result list, and the right-hand block shows a typical argument list.

6.4.2 TCP/IP

In the RSPE, the used communication protocol between the SCF and the SSF, SRF and SDF is the Transmission Control Protocol/Internet Protocol (TCP/IP). This protocol enables any network computer to communicate with any other network computer. As a result, all computers appear to exist in the same network (i.e. the internet).

Figure 6-6 shows the TCP/IP architecture mapped to the OSI Reference Model.
The IP protocol is the main part of TCP/IP. It provides a connectionless data transfer service, and moves packets from one computer to another through the internet. The other protocols in the network layer assist the IP protocol.

On the transport layer two protocols are located. The UDP protocol is a simple connectionless, best-effort protocol that adds little to the underlying IP service. The TCP protocol is connection-oriented and provides a reliable transfer of sequenced data streams.

Some application protocols and services are:

- **Packet InterNet Groper (Ping)** - Can be used to test connectivity between two computers on the internet.

- **Telnet** - Allows a user to login on another computer on the internet.

- **File Transfer Protocol (FTP)** - Allows a user to transfer data between a local computer and a remote computer in both directions.

- **Simple Mail Transfer Protocol (SMTP)** - Can be used to transfer electronic mail messages to other computers.

In the RSPE, none of these application layer protocols are used. Instead, Erlang is used to generate messages.

For more information about TCP/IP refer to [MART94]. The Erlang messages that are available in the RSPE, and can be transmitted via TCP/IP, will be described in the following section.
6.5 Erlang messages

As mentioned before, Erlang messages can be used in the RSPE for the SCF-SSF, SCF-SDF and SCF-SRF communication. The messages that are available, will be discussed in the next subparagraphs.

6.5.1.1 SCF-SSF Messages

The SCF-SSF communication is mainly used to initiate and end IN service execution, and to make some user interaction possible. The latter actually is more a task for the SRF rather than the SSF, but some interaction capabilities have been implemented anyway.

The SCF-SSF messages are described in Table 6-1.

Table 6-1: SCF-SSF messages

<table>
<thead>
<tr>
<th>Direction</th>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSF to SCF</td>
<td>Initiate</td>
<td>Issued by the SSF on encountering a static breakpoint in basic call processing.</td>
</tr>
<tr>
<td></td>
<td>Provide</td>
<td>After issuing an Initiate message, the Provide message is sent to provide the SCF with data associated to the static breakpoint.</td>
</tr>
<tr>
<td></td>
<td>Event</td>
<td>Issued by the SSF on encountering a dynamic breakpoint.</td>
</tr>
<tr>
<td></td>
<td>Simulation_end</td>
<td>Used to clean up pending non call related (ncr) events after a Transfer message.</td>
</tr>
<tr>
<td>SCF to SSF</td>
<td>Continue</td>
<td>Issued by the SCF to have the SSF resume basic call processing.</td>
</tr>
<tr>
<td></td>
<td>Dtmf receive</td>
<td>Receives DTMF tones from a user-terminal.</td>
</tr>
<tr>
<td></td>
<td>Dtmf send</td>
<td>Sends a list of digits as DTMF tones over the indicated channel.</td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td>Displays a prompt on a digital terminal and returns the user input.</td>
</tr>
<tr>
<td></td>
<td>Join</td>
<td>Adds a party to a channel in multi-party services.</td>
</tr>
<tr>
<td></td>
<td>Load data</td>
<td>Changes call related data in the SSF, including the state of the Basic Call Model (BCM) process.</td>
</tr>
<tr>
<td></td>
<td>Message</td>
<td>Displays a message on a digital terminal.</td>
</tr>
<tr>
<td></td>
<td>Set monitor</td>
<td>Sets a dynamic breakpoint on a state of the BCM process.</td>
</tr>
<tr>
<td></td>
<td>Split</td>
<td>Removes a party from a channel in multi-party services.</td>
</tr>
<tr>
<td></td>
<td>Start tone</td>
<td>Puts a tone on the indicated channel.</td>
</tr>
<tr>
<td></td>
<td>Stop tone</td>
<td>Stops a previously started tone.</td>
</tr>
<tr>
<td></td>
<td>Transfer</td>
<td>Indicates to the SSF that service processing in the SCF ended and that the SSF can continue basic call processing.</td>
</tr>
<tr>
<td></td>
<td>Unset monitor</td>
<td>Unsets a previously set dynamic breakpoint.</td>
</tr>
</tbody>
</table>
6.5.1.2 SCF-SDF Messages

The Service Data Function (SDF) is the entity which is used to store all user related information. It can be seen as a central database, which can be accessed by the SCF. Before the SCF can send or receive user information, it should know the respective user ID. By providing this user ID to the SDF, the proper data can be accessed. Figure 6-7 shows what the data structure of a single user profile looks like.

As can be noticed from the figure, the user ID gives access to a tree-like data structure, which, among others, contains user data. In the context of this project, only the user data is important. The user data contains the complete user profile. The user profile can be read or written all at once, or tag by tag. In the figure two tags are shown: algorithm and numbers. Besides the tag values, the tag types are specified in the user profile as well. To get information from the SDF and to add or update information, some SCF-SDF messages have been implemented in the RSPE.

Table 6-2 shows the SCF-SDF messages.

<table>
<thead>
<tr>
<th>Direction</th>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCF to SDF</td>
<td>Classes</td>
<td>Returns the names of the profiles stored in the SDF.</td>
</tr>
<tr>
<td></td>
<td>Erase</td>
<td>Erases some data in the SDF.</td>
</tr>
<tr>
<td></td>
<td>Erase all</td>
<td>Erases all data in the SDF.</td>
</tr>
<tr>
<td></td>
<td>Get</td>
<td>Gets some data from the SDF.</td>
</tr>
<tr>
<td></td>
<td>Get all</td>
<td>Gets all data from the SDF.</td>
</tr>
<tr>
<td></td>
<td>Get sdf</td>
<td>Returns the address of the SDF.</td>
</tr>
<tr>
<td></td>
<td>Put</td>
<td>Puts data in the SDF.</td>
</tr>
<tr>
<td></td>
<td>Sdfnode</td>
<td>Returns the node on which the SDF is located.</td>
</tr>
</tbody>
</table>
6.5.1.3 SCF-SRF Messages

The SRF is used to make user interaction possible. The SRF can play announcements or collect information from the user. A complete script can be specified and executed as well. The SCF initiates certain SRF actions or scripts. Subsequently, the SRF returns a result or collected information from the user. To enable this communication, several SCF-SRF messages have been implemented in the RSPE.

In Table 6-3, the SCF-SRF Erlang messages are described.

<table>
<thead>
<tr>
<th>Direction</th>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCF to SRF</td>
<td>Play announcement</td>
<td>Used to have the SRF play an announcement.</td>
</tr>
<tr>
<td></td>
<td>Prompt and collect</td>
<td>Used to have the SRF prompt for input and returns the user input.</td>
</tr>
<tr>
<td></td>
<td>Run script</td>
<td>Used to have the SRF run a script with a certain script ID.</td>
</tr>
<tr>
<td>SRF to SCF</td>
<td>Collected info</td>
<td>Sends the collected information to the SCF.</td>
</tr>
<tr>
<td></td>
<td>Load script</td>
<td>Ask the SCF to download a certain script to the SRF.</td>
</tr>
<tr>
<td></td>
<td>Result</td>
<td>Returns the result of a Run script message to the SCF.</td>
</tr>
<tr>
<td></td>
<td>Run result</td>
<td>Used to have the SCF store the result data in a certain location.</td>
</tr>
<tr>
<td></td>
<td>Run scf script</td>
<td>Used to have the SCF run a script with a certain script ID.</td>
</tr>
<tr>
<td></td>
<td>Specialized resource report</td>
<td>Sends a specialized resource report to the SCF.</td>
</tr>
</tbody>
</table>

6.6 Summary and Conclusions

In this chapter, the IN/CC environment has been discussed. A general IN/CC architecture has been compared with the IN/CC architecture used for prototyping purposes. The main difference between the two architectures is the used communication protocols between the different network entities. In the general architecture the INAP protocol is used. In the prototyping architecture, operations are exchanged as Erlang messages, possibly over the TCP/IP protocol. Both protocols have been described briefly. Subsequently, the defined Erlang messages have been discussed for the SCF-SSF, SCF-SDF and SCF-SRF communication.

This chapter forms a good starting-point for the following chapter, which will focus on the prototyping of IN/CC services.
7. Implementation

In this chapter the implementation of the IN/CC environment will be discussed. New software needs to be developed both at the PC side and at the IN side of the environment. This chapter focuses on this software. We will also take a look at the implemented services and features and the corresponding information flows.

7.1 Introduction

In the previous chapter the IN/CC environment has been discussed. The architecture of the network and the used protocols have been considered. The available Erlang messages have been described as well. This information is a good starting-point for the actual implementation of the IN/CC environment.

At the PC side, some software is necessary to enable the PC to communicate with the chipcard and with the modem. Further, a user interface is needed to ease profile management.

For the IN side, new software needs to be developed as well. First of all, software is necessary to setup a connection between the modem at the PC side and the modem at the IN side. This is done by the SCF after a certain IN service number has been called by the user. The software enables the SRF to understand the messages received from the PC and to send the proper messages to the PC. Further, it handles communication between the SRF and the SDF, via the SCF, so user data and service profiles can be transferred between the SDF and the SRF.

Before we will consider the software, the used technology for the prototyping at the PC side will be briefly described.

7.2 Used Technology

This paragraph briefly describes the used chipcard terminal, chipcards and modems.

7.2.1 Chipcard Technology

The used chipcard terminal in the prototyping environment is the ChipDrive Extern from Towitoko Electronics. It is a universal chipcard terminal which can be used to read all kinds of ISO 7816 compatible chipcards. The chipcards that are used for the prototyping purposes are memory cards. The memory cards do not contain a processor, but only a 8 Kbyte EEPROM. The cards can be accessed using the so called I2C protocol, which is a synchronous protocol. In synchronous transmission a linear relation between the bit rate on the I/O line and the clock frequency on the CLK pin exists.

A Dynamic Link Library (DLL) is available, which can be used to develop software for the chipcard terminal. The DLL provides functions to initialize the terminal and to write data to and read data from the card.
7.2.2 Modem Technology

The used modems in the prototype environment support the V.42 protocol. This protocol, also referred to as Link Access Procedure for Modems (LAPM), has the following characteristics:

- Provides error detection by using Cyclic Redundancy Check (CRC).
- Provides error correction by using automatic retransmission of data after a CRC check has failed.

The V.42 protocol packetizes the data to be transferred. It adds a CRC check field to the packet, so the receiver can check whether or not the packet has been transferred without errors. The packetizing is done transparently to the user.

The underlying protocol is V.32bis which supports a transmission rate of 14400 Bps. It is beyond the scope of this report to further discuss the modem protocols. For more information refer to [V.32bis] and [V.42].

7.3 Profile Management Software

In this paragraph the software that has been developed for the PC will be considered. The software has the following capabilities. It can be used to:

- Edit profiles.
- Upload profiles to and download profiles from the IN.
- Write profiles to and read profiles from a chipcard.

To achieve the above mentioned capabilities, the following four software entities are necessary:

- User Interface - Makes editing profiles possible and enables users to initiate uploading, downloading, reading and writing profiles.
- Modem Driver - Enables communication with the modem at the PC side of the IN/CC environment.
- Terminal Driver - Enables communication with the chipcard terminal.
- Parser - Can be used to convert C structs to Erlang terms and Erlang terms to C structs. The format of Erlang terms will be discussed later.

The hierarchical structure of these software entities is shown in Figure 7-1.
From the User Interface, functions of the Terminal Driver, Modem Driver and Parser can be called. The most important functions have been shown in the figure.

The software can run in two different modes:

- **Service Oriented** - For every IN service a *service profile* exists, which contains user defined parameters.

- **User Oriented** - For every user a *user profile* exists (also called a flexible service profile), which completely describes his personal IN services and the user defined parameters. The user profile describes which services the user can use, how these services work (the service logic) and the user defined parameters.

So in the service oriented mode, one profile is needed for every service that the user wants to use. The profile only specifies the values of the user definable parameters. In the user oriented mode, only one profile is needed. It does not only contain the values of the user definable parameters for all services, but it also contains the so called *service logic* for every service, which describes the Service Independent Building blocks (SIBs), their parameters, outlets and the way the SIBs are connected to each other. So a user profile provides much more information than a collection of service profiles. This however is also a disadvantage of the user oriented mode, because much more information needs to be handled. This not only increases transmission times, but it also requires more chipcard memory space. The latter disadvantage is the reason why in the prototype software the storage of service logic has not been implemented, but the software can easily be adapted to support this feature as well.

In the following sections both modes will be described. However, at the IN side only the user oriented mode is supported at this moment. The service oriented mode might be implemented in the future. This mode will be considered in the next section, but the remainder of the report focuses on the user oriented mode only.
7.3.1 Service Oriented Mode

In this subparagraph the service oriented mode will be discussed. The User Interface is very flexible and does not need to be updated whenever existing IN services have been changed or when new IN services have been introduced. The only thing that must be changed in these situations, is the so called Service Profiles Descriptor (SPD). The SPD can be downloaded from IN. The SPD describes the structure of the service profiles. In other words, it defines exactly what kind of parameters are expected to be specified by the user. Before the user can edit the service profiles, the SPD must be downloaded. When the SPD has been downloaded, the PC software is able to ask the user for the proper parameters with the respective parameter types. The SPD will be discussed in more detail in the next section.

7.3.1.1 Service Profiles Descriptor

The structure of the SPD is shown in Figure 7-2. As shown, for every IN service the following information is provided by the SPD:

- Service names
- Service IDs
- Tag names
- Tag types

![Service Parameters Descriptor](image)

Every service can be described with one or more tags. A tag is a field which can contain a certain user definable parameter or list of parameters. For every tag, a tag name and a tag type is defined. The following tag types exist:
Chipcards in Intelligent Network Services

- **numbers** - Tag can only contain alphanumeric characters (0-9).
- **list_of_numbers** - Tag can contain one or more numbers.
- **digits** - Tag can contain a telephone number containing alphanumeric characters and the '*', '#', and '-' character.
- **list_of_digits** - Tag can contain one or more telephone numbers.
- **string** - Tag can contain a string.
- **list_of_strings** - Tag can contain one or more strings.

In the figure only two services have been shown with two respectively three tags. Of course the SPD can describe more services and the number of tags can vary among different services.

The SPD structure shown in Figure 7-2 is a schematic representation of the SPD. In reality, the SPD is described somewhat different. The original SPD resides in the IN and on request is sent from the SRF to the PC. The SPD is sent as an Erlang term, which has the following format:

```plaintext
{ServiceName1,ServiceID1,[{TagName1,TagType1},{TagName2,TagType2},...]}{ServiceName2,ServiceID2,[{TagName1,TagType1},{TagName2,TagType2},...]}{ServiceName3,ServiceID3,[{TagName1,TagType1},{TagName2,TagType2},...]}...
```

When the SPD is received by the PC, it will be converted to a C struct, so the software can use the provided description of the service profiles. For more information about Erlang terms refer to [ARM93].

To clarify the use of the SPD an example will be given. The Call Distribution service could be described as depicted in Figure 7-3.

![Service Parameters Descriptor](Figure 7-3: Description of the Call Distribution service)
In the first place, the service name and an unambiguous service ID is provided. Further, the SPD indicates to the software, that the user should specify which distribution algorithm should be used (the specified value should be of the type ‘numbers’), a list of telephone numbers to which calls should be distributed (a list of digits is expected), and a list of numbers that specifies the priority or percentage (dependent on the used distribution algorithm) of every telephone number in the former list.

Once the SPD has been downloaded to the PC it will be stored on the hard disk. Every time the software is executed, the SPD will be loaded from the hard disk automatically, so the user does not have to download the SPD every time he uses the software. Right now, no SPD version management has been implemented. If the service oriented mode will be used more often in the future, the user should be notified when a new version of the SPD is available.

7.3.1.2 Structure of transferred Service Profiles

Service profiles can be downloaded from and uploaded to the IN. The service profiles are transferred between the PC and the SRF as Erlang terms. Erlang terms describing service profiles may have the following format:

```erlang
{Service/d1 ,[[tag1_item1 ,tag1_item2 ],[tag2_item1 ],[tag3_item1 ,tag3_item2 ,tag3_item3 ]]} 
{Service/d2 ,[[tag1_item1 ],[tag2_item1 ],[tag3_item1 ,tag3_item2 ,tag3_item3 ]]} 
{Service/d3 ,[[tag1_item1 ,tag1_item2 ,tag1_item3 ],[tag2_item1 ,tag2_item2 ]]} 
... 
```

These Erlang terms specify the service profiles of three services with a different number of tags per service and a different number of items per tag. The first service for instance, has three tags, with two, one respectively three items.

7.3.1.3 Structure of Service Profiles on a Chipcard

As mentioned before, the software on the PC can be used to write service profiles to and read service profiles from a chipcard. Except for the service profiles, a simple user profile can be stored on the card as well. The content of the chipcard is depicted in Figure 7-4.
As shown, the user profile consists of a user name, a user ID and a PIN. The service profiles have a more complex structure. First of all, the number of service profiles on the card is indicated. Further, for each service on the card, a service profile consisting of the following elements has been stored:

- **Service ID** - Unambiguously identifies the service.
- **Number of tags** - Specifies the number of tags that have been defined for the service.
- **Number of items** - Specifies the number of items for every tag.
- **Table with values** - Specifies the user defined values of every item for every tag.

Note that the number of items may vary among the different tags. The figure only shows two service profiles, but of course the number of service profiles is variable.

To clarify the structure of stored service profiles, an example is given in Figure 7-5. Suppose two service profiles have been stored on the card. The first service is the Call Distribution service. Three telephone numbers have been specified with their respective priorities. The used priority distribution algorithm is indicated with a ‘2’. In section 7.3.1.1 the SPD for the CD service was given.

The second service is the Call Abbreviation service. Its service profile contains three telephone numbers and their respective abbreviations.
7.3.2 User Oriented Mode

In the user oriented mode no SPD is necessary. The user profile itself contains information about the tag names and tag types. However, no service names and service IDs are provided by the profile. The tag names should make clear to which service the tags apply. For this mode the same parameter types have been defined as for the service oriented mode (i.e. string, list of strings, numbers, list of numbers, digits, list of digits).

As mentioned before, at this moment storage of the service logic is not supported by the software. The only thing that can be managed by the software is just a part of the user profile. This part contains the following information:

- Tag names
- Tag types
- Tag values

The information can be transferred between the PC and the SRF and between the PC and a chipcard. In both situations, the data is transferred as an Erlang term:

```
[{{tagname1,tagtype1,tagvalue1}},{tagname2,tagtype2,tagvalue2},...]
```

For every user definable parameter the tag name and tag type have been specified. Additionally, tag values must be specified by the user. The format of a tag value is dependent on the respective tag type.
Table 7-1 shows examples of tag values for the different tag types.

<table>
<thead>
<tr>
<th>Tag type</th>
<th>Example of tag value</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>&quot;James Bond&quot;</td>
</tr>
<tr>
<td>list_of_strings</td>
<td>[&quot;string1&quot;,&quot;string2&quot;,&quot;string3&quot;]</td>
</tr>
<tr>
<td>numbers</td>
<td>[1,2,3,4,5,6]</td>
</tr>
<tr>
<td>list_of_numbers</td>
<td>[[1,2,3],[4,5,6],[7,8,9,0]]</td>
</tr>
<tr>
<td>digits</td>
<td>[*2,1]</td>
</tr>
<tr>
<td>list_of_digits</td>
<td>[[0,1,2,3,4],[#7,8,9]]</td>
</tr>
</tbody>
</table>

As shown, numbers and digits are stored as an Erlang list. The reason for this format is that numbers and digits are stored like this in the RSPE. So when a user specifies that his telephone number is 016249022, this will be stored as [0,1,6,2,4,9,0,2,2].

7.4 IN/CC Information Flows

This section discusses the information flows for typical profile service communication sessions, like upload, download and disconnect. Other relevant information flows will be discussed later.

For profile service operation the following phases can be identified:

1. **Initiation:**
   - The user calls a service number (e.g. 0900 number).
   - The SSF identifies the requested service.
   - The SSF hands the control over to the SCF.

2. **Identification:**
   - The card sends the user ID to the SRF.
   - The SRF sends the user ID to the SCF.

3. **Command:**
   - The SRF ask the user for a command (i.e. upload, download or disconnect).
   - The PC software returns a command.

4. **Upload/download:**
   - The service profiles are uploaded/downloaded to/from the SDF via the SCF and SRF.

5. **Termination:**
   - The SCF hands the control over to the SSF.
   - The connection is closed.
Figure 7-6 shows the information flow for profile services.

The figure shows how the SCF gets the user ID and the command from the user side. Dependent on the command a download, upload or disconnect session is executed. Besides the three mentioned commands, some other commands can be issued by the user interface. These will be discussed in the following paragraph.
7.5 Implemented IN/CC Services

To be able to demonstrate the use of IN/CC services, some have been implemented in the RSPE. One of them is a typical profile service (Abbreviated Dialing service), while the other two could be called secure services (Account Card Calling and Universal Personal Telecommunications). The three services will be discussed in the next sections.

7.5.1 Account Card Calling Service

From the user interface an Account Card Calling (ACC) session can be initiated. The user will be asked to enter the terminating telephone number and subsequently, the software will use the modem to setup a call to the specified destination. The modem will not directly call the terminating number, but a certain IN service number will be called. At the IN side, a service script will be started which asks for the user ID. The SCF uses this ID to access user specific information in the SDF and it will check whether or not the ACC service is enabled for this user. If so, the modem at the PC side will hand over the access to the telephone line to the connected telephone (refer to Figure 6-3) and the IN script disconnects the SRF and transfers the call to the appropriate destination (i.e. the specified terminating number). So from that moment, data will no longer be transferred between the two modems, and the connection can be used for voice traffic. Also some charging mechanism is activated to administrate the costs of the call. All charging information will be stored in the SDF.

Figure 7-7 shows the information flow between the different network entities.

The upper part (above the dashed line) of the figure is equal to the upper part of Figure 7-6. However, in this case the command is ‘Call’. After the command has been issued, the number to be called is fetched. Subsequently, the acc_enabled tag is read from the SDF to verify whether or not the service is enabled for the user. If so, the SRF is disconnected from the SSF, the number to be called is sent to the SSF and the continue operation gives the control over the call back to the SSF. Finally, a connection with the B-party is established.
7.5.2 Universal Personal Telecommunications Service

The user interface provides the capability to perform UPT (Universal Personal Telecommunications) registration. The user can specify on which terminal he wants to be registered and subsequently, the software sends the user ID to the IN and the number of the specified telephone. The IN takes care of the actual registration process.

Once the user is registered, he can be called by dialing 111 followed by his user ID. The call will be routed to the proper location. Also, he can use his personal services from that terminal, like the abbreviated dialing service. Figure 7-8 shows the UPT registration information flow.
Again the upper part of the figure resembles Figure 7-6, however in this case the command is 'Register'. The SCF asks the SRF to get the number of the telephone on which the user wants to be registered. This number is returned to the SCF and stored in the SDF. From that moment the user is registered.

### 7.5.3 Abbreviated Dialing Service

The Abbreviated Dialing service provides users with the capability to dial abbreviated telephone numbers. The user should provide a list with original numbers and a list with the corresponding abbreviations. Once these lists are uploaded to the IN (to the SDF), the service can be used. The user should dial 222 (the prefix) followed by the abbreviation. The prefix is necessary to indicate to the IN that the Abbreviated Dialing service is being used. It initiates the execution of an IN service script that accesses the user's database, translates the abbreviated number to the original number and forwards the call to the proper destination. Since the user has been registered as an UPT user, his user ID is already known and it is not necessary to get the user ID from the chipcard again. Figure 7-9 shows the information flow between the different network entities.
When the user dials the abbreviated dialing prefix 222 and the abbreviated number, the SSF recognizes this as an IN call and sends an Initial DP to the SCF. This message includes, among other information, the user ID and the dialed number. The SCF uses the user ID to access the SDF and receives a list with original numbers and a list with abbreviated numbers. The latter is searched and if the dialed abbreviated number is found in the list then the corresponding original number is returned to the SSF. Finally, the SCF hands over the control over the call to the SSF by sending the Continue operation and a connection with the original number is established.

7.6 Additional Features

Some additional features that have been implemented in the user interface will be discussed briefly in this section.

7.6.1 Card Personification

The software provides the capability to personify new, empty chipcards. The user can store, besides his user profile (name, user ID and PIN), a photograph on the chipcard. The photograph must be compressed using the so called JPEG algorithm, since this compression technique is the most efficient at this moment. More information about JPEG can be found in Appendix A: JPEG Compression. If a photograph is stored on the user’s chipcard, it will be shown on the screen as soon as the card is inserted in the terminal. This feature is mainly intended to provide some additional authentication.

---

1 Also on the Internet a lot of information on JPEG is available, for instance: http://www.faqs.org/faqs/jpeg-faq/
7.6.2 Charging Information

As mentioned before, when using the Account Card Calling service, the user is being charged and the charging information is stored in the SDF. Also when the Abbreviated Calling service is used or when the user interface is used to communicate with the IN, charging information is stored in the SDF. The content of the charging information is determined in the SSF as soon as the A party is disconnected. Subsequently, the SCF sends a request for the charging data to the SSF, which returns the requested information to the SCF. The SCF gets the old charging information from the SDF, adds the new charging data just received from the SSF to the old data and stores the result in the SDF. The information flow for the charging procedure is shown in Figure 7-10.

![Figure 7-10: Charging procedure](image)

The user interface provides the capability to download this charging information from the SDF. After the user ID has been sent to the IN, a detailed overview of the latest calls will be downloaded from the SDF and shown on the screen. The information is transferred as an Erlang term. The term, the so called charging profile, has the following structure:

```prolog
[{parametername1,{parametertype1,parametervaluel1}},
 {parametername2,{parametertype2,parametervalue2}},...]
```

This structure is similar to the data structure in the user oriented mode. In this way, the content of the charging information is flexible. At the IN side of the system, the structure of the charging data can be changed any time, without changing the user interface software is required. The charging profile completely describes the transferred parameters, their types and their values. Table 7-2 shows the current content of the charging information. The values in the table are just given as an example.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>a number</th>
<th>b number</th>
<th>c number</th>
<th>date</th>
<th>duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>list of strings</td>
<td>list of strings</td>
<td>list of strings</td>
<td>list of strings</td>
<td>list of strings</td>
</tr>
<tr>
<td>Values</td>
<td>“0169022”</td>
<td>“0162582”</td>
<td>-</td>
<td>“26/11/97”</td>
<td>“00:03:12”</td>
</tr>
<tr>
<td></td>
<td>“0169022”</td>
<td>“0169422”</td>
<td>111007</td>
<td>“28/11/97”</td>
<td>“00:17:53”</td>
</tr>
</tbody>
</table>

Note that the a_number specifies the originating numbers of the calls and the b_number specifies the terminating numbers of the calls. The c_number is used to indicate that the B party is called by using the UPT service.
Figure 7-11 shows the information flow for a charging information download session. It resembles a normal download session. After the SCF gets the Charging_info command, the charging information of the respective user is received from the SDF and sent to the PC. If the information is received by the PC, it returns an acknowledgment.

Before the IN software will be discussed, we will take a closer look at the IN/CC communication messages transferred between the different entities.

### 7.7 Communication Messages

In this paragraph some attention will be paid to the used messages between the different entities in the IN/CC environment. Several messages have already been discussed in the previous chapter. In this paragraph however, some new messages will be defined.

#### 7.7.1 PC-SRF Communication

In this section, we will take a more detailed look at the communication between the PC and the SRF. The relevant messages in this context are those which are transferred between the PC and the SRF in Figure 7-6 to Figure 7-11. These messages are summarized in Table 7-3.
### Table 7-3: PC-SRF messages

<table>
<thead>
<tr>
<th>Direction</th>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRF to PC</td>
<td>Ack_auth_code_card</td>
<td>Used to acknowledge the authentication code received from the card. Now the PC software knows that the card is authenticated.</td>
</tr>
<tr>
<td></td>
<td>Get_userid</td>
<td>The SRF asks for a certain ID, which unambiguously identifies the user.</td>
</tr>
<tr>
<td></td>
<td>Get_command</td>
<td>The SRF asks the user interface for a command. The following commands can be issued by the user interface:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Upload - Uploads user profile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Download - Downloads user profile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Register - Registers user on the specified location</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Call - Initiates the Account Card Calling service</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Charging_info - Downloads charging information</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Disconnect - Disconnect the PC from the IN</td>
</tr>
<tr>
<td></td>
<td>Get_number</td>
<td>The SRF asks the user interface for the number to be dialed in the Account Card Calling service.</td>
</tr>
<tr>
<td></td>
<td>Get_originating_number</td>
<td>The SRF asks the user interface for the number to be registered on in the Universal Personal Telecommunications service.</td>
</tr>
<tr>
<td>PC to SRF</td>
<td>Ack_auth_code_srf</td>
<td>Used to acknowledge the authentication code received from the SRF. Now the IN software knows that the SRF knows that it has been authenticated.</td>
</tr>
<tr>
<td></td>
<td>Ack_charging_info</td>
<td>The user interface acknowledges the received charging information.</td>
</tr>
<tr>
<td></td>
<td>Ack_download</td>
<td>The user interface acknowledges the received service profile.</td>
</tr>
</tbody>
</table>

### 7.7.2 SCF-SRF Communication

As shown in Figure 7-6 to Figure 7-11, some new messages between the SCF and SRF are necessary. The messages that had not been implemented yet, are summarized in Table 7-4.

### Table 7-4: SCF-SRF messages

<table>
<thead>
<tr>
<th>Direction</th>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCF to SRF</td>
<td>Get_userid</td>
<td>The SCF asks the SRF to get the user ID from the PC software.</td>
</tr>
<tr>
<td></td>
<td>Get_command</td>
<td>The SCF asks the SRF to get a command from the PC software.</td>
</tr>
<tr>
<td></td>
<td>Get_number</td>
<td>The SCF asks the SRF to get the number to be dialed in the Account Card Calling service from the PC software.</td>
</tr>
<tr>
<td></td>
<td>Get_originating_number</td>
<td>The SCF asks the SRF to get the number to be registered on in the Universal Personal Telecommunications service from the PC software.</td>
</tr>
<tr>
<td></td>
<td>Upload_profiles</td>
<td>The SCF asks the SRF to initiate a service profiles upload session.</td>
</tr>
<tr>
<td>SRF to SCF</td>
<td>Ack_download</td>
<td>The SRF acknowledges a successful download session.</td>
</tr>
<tr>
<td></td>
<td>Ack_charging_info</td>
<td>The SRF acknowledges a successful charging information download session.</td>
</tr>
</tbody>
</table>
7.7.3 SCF-SSF Communication

To obtain charging information from the SSF, a new operation has been introduced in the SCF (refer to Figure 7-10). Table 7-5 shows this operation.

<table>
<thead>
<tr>
<th>Direction</th>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCF to SSF</td>
<td>Get_calldata</td>
<td>The Get_calldata operation takes a parameter to indicate what information from the call data is requested. In this case the parameter should be 'charging_data' to obtain the charging information from the SSF.</td>
</tr>
</tbody>
</table>

7.8 IN Software

At the IN side of the IN/CC environment, the software can be described by Service Independent Building Blocks (SIBs) and scripts. As explained before, a SIB is a reusable function which can be used in different services. A script is formed by linking SIBs together and it completely specifies a user profile or a service profile.

This paragraph discusses the most important SIBs and scripts that have been used in the IN/CC prototype.

7.8.1 Service Independent Building Blocks

Before the used SIBs will be discussed, we first will take a brief look at a typical representation of a SIB. Figure 7-12 shows this representation.

![Figure 7-12: Representation of a SIB](image)

The shown figure is almost equal to Figure 2-2. New in the figure is the 'Event' arrow at the left side. It indicates that the SIB is re-entrant. This means that the SIB will be re-entered on a certain event. Table 7-6 shows important SIBs that have been used in the prototype. These SIBs already existed before this project was started, and they have been reused. The PNC40 SIB has been extended with code to enable the required modem-to-modem communication.

Table 7-7 shows all new SIBs that have been developed especially for this project.
Table 7-6: Reused Service Independent Building Blocks

<table>
<thead>
<tr>
<th>SIB Representation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CTR10</strong>&lt;br&gt;<img src="image1.png" alt="CTR10 Diagram" /></td>
<td>The CTR10 SIB is the Connect To Resource SIB. It can be used to connect the IP to the user. Firstly, it sends the load_data message to the SSF, which contains the information needed to connect the IP and the user. Subsequently, the ok or nok outlet is selected. The SIB waits for an assist_request_instructions (ari) event, which is generated when the SCF receives an ari message from the IP. Then the connection between the IP and the user is established and the ari outlet is selected.</td>
</tr>
<tr>
<td><strong>DFC10</strong>&lt;br&gt;<img src="image2.png" alt="DFC10 Diagram" /></td>
<td>The DFC10 SIB is the Disconnect Forward Connection SIB. It sends a load_data message to the SSF to disconnect the IP from the user. The ok or nok outlet is selected, depending on the result of the load_data.</td>
</tr>
<tr>
<td><strong>PNC10</strong>&lt;br&gt;<img src="image3.png" alt="PNC10 Diagram" /></td>
<td>The PNC10 SIB is the Prompt and Collect SIB. It can be used to send information to the user and subsequently receive information from the user. The four parameters that must be provided, indicate the type and contents of the message to be sent, the type of information to be received, whether or not the IP is allowed to release the connection after it received the required information from the user, and the destination of the received information. After the information has been sent to the user, the ok or nok outlet is selected. Then the SIB waits for a collected_info event, stores the information and selects the result outlet.</td>
</tr>
<tr>
<td><strong>PNC40</strong>&lt;br&gt;<img src="image4.png" alt="PNC40 Diagram" /></td>
<td>The PNC40 SIB is the Prompt and Collect SIB. Unlike the previous three SIBs, this SIB is not a SCF building block, but a SRF building block. It seizes the proper IP resource, connects it to the incoming channel, sends and receives some requested information, disconnects the resource from the channel, and finally releases the resource. The ok or nok outlet is selected after information is sent to the user. Then the SIB waits for a stop or result event, if necessary stores the result and selects the complete outlet.</td>
</tr>
</tbody>
</table>
Table 7-7: New Service Independent Building Blocks

<table>
<thead>
<tr>
<th>SIB Representation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbr_number&lt;br&gt;orig_number&lt;br&gt;user_id</td>
<td>The Abbr_dialing10 SIB reads two lists of numbers from the SDF by using the specified user_id. One list contains original telephone numbers and the other list contains the respective abbreviations. The SIB searches the latter list for a match with abbr_number. If a match is found, the corresponding original number is returned in orig_number.</td>
</tr>
<tr>
<td>user_id&lt;br&gt;sdf_data</td>
<td>The Charging_info10 SIB uses the specified user_id to access the SDF. It reads the charging information of the respective user and stores this information in sdf_data.</td>
</tr>
<tr>
<td>user_id&lt;br&gt;sdf_data</td>
<td>The Download10 SIB reads the user profile of the user indicated by user_id from the SDF. Subsequently, the SIB removes all tags, their tag types and their values from the received Erlang term, which are not supposed to be sent to the user interface. Finally, the resulting Erlang term is stored in sdf_data.</td>
</tr>
<tr>
<td>SIB Representation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Get_call_data10</td>
<td>The <code>Get_call_data10</code> SIB takes the tag as a parameter. It indicates which part of the call data in the SSF is required by the SCF. In this project the SIB is used to request charging information from the SSF. The tag should be 'charging_data' in this case. The SIB sends a <code>get_call_data</code> operation to the SSF, which in turn returns the requested information. The result is stored in the location indicated by <code>where_to_put_result</code>.</td>
</tr>
<tr>
<td>Register_incc10</td>
<td>The <code>Register_incc10</code> SIB takes an A number (<code>anr</code>) as a parameter. If necessary the user, indicated by <code>fspid</code>, is being unregistered from the location where he is registered. Subsequently, he is registered to the location specified by <code>anr</code>. The specified location is being stored in the SDF, together with the address of the SSF and SCF. The <code>type</code> parameter is not being used at the moment.</td>
</tr>
<tr>
<td>Store_charging10</td>
<td>The <code>Store_charging10</code> SIB uses the <code>user_id</code> to access the SDF and receive the user's old charging information. Then it adds the <code>charging_data</code> to the old charging data and stores the result in the SDF. The SIB only keeps the charging information for the latest nine calls and discards the remaining data. To give the reader an idea of what a typical SIB looks like, the Erlang code of the <code>Store_charging10</code> SIB is shown in appendix B.</td>
</tr>
<tr>
<td>Upload10</td>
<td>The <code>Upload10</code> SIB takes the user profile received from the user interface as a parameter (<code>sdf_data</code>). The values of the received tags are being updated in the SDF, while the remaining tags in the SDF are not being touched.</td>
</tr>
</tbody>
</table>
7.8.2 Scripts

The SIBs that just have been described have been used to build several scripts in the RSPE. Scripts are needed both on the SCF and on the SRF. The scripts for both entities will be described in this subparagraph.

7.8.2.1 SCF Scripts

For this project two new SCF scripts have been designed. The first script is used to enable communication with the user interface at the PC side of the system. It is being started at the SCF when a certain service number (0646) is being dialed. The SSF will recognize this number and hand the control over the call over to the SCF. In the SCF the script depicted in Figure 7-13 is executed.

After the start10 SIB has been executed, a monitor is set on the SSF to detect a disconnect event. Next, the ctrlO SIB (refer to Table 7-6) is started. From that point the SRF is connected to the user interface and information can be transferred between the two entities. Firstly, a pnc10 SIB is executed with the 'setup' parameter. This parameter causes the SRF modem to be initialized and to go off-hook. If both modems have been connected to each other, the SRF sends a 'Get_userid' request and the PC software returns the user ID as a response. In the next pnc10 SIB, the SRF sends a 'Get_command' message.

Note that the pnc10 SIB initiates the execution of a script on the SRF. This script will be discussed in section 7.8.2.2.

After the 'Get_command' message is sent, the pnc10 SIB waits for one of the following commands from the user interface:

- **Download** - As a result the download10 SIB is executed which reads the user profile from the SDF. Subsequently, the profile is transmitted to the user interface by executing the pnc10 SIB.

- **Upload** - The pnc10 SIB causes the user profile to be received from the user interface. Next, the upload10 SIB stores the profile in the SDF.

- **Call** - Initiates the Account Card Calling service. The pnc10 SIB gets the terminating number of the call. The interrupt10 SIB forces the SSF in a special state, the so called scf_interrupt state at the originating side and the t_scf_interrupt state at the terminating side. This makes the SSF wait for instructions from the SCF. This is necessary, since the download10 SIBs need to send some information to the SSF. The compare10 SIB checks whether or not the ACC service is enabled for the user. If it is enabled, the second compare10 SIB checks if the terminating number has a prefix '111', which indicates that it is a UPT call. If not, the loaddata10 SIB loads the specified terminating number to the SSF. If so, the prefix is removed in the modify10 SIB and the proper terminating number is loaded to the SSF in loaddata10. When the done10 SIB is reached, the A party is disconnected from the SRF and connected to the specified terminating number.
- **Register** - The pnc10 SIB is executed and the telephone number on which the user wants to be registered is acquired. The register_inc10 SIB takes care of the actual registration and stores the location of the user in the SDF.

- **Charging info** - The charging_info10 SIB gets the charging data of the user from the SDF. Subsequently, the data is transmitted to the user interface by executing the pnc10 SIB.

- **Disconnect** - Causes the connection between the SRF and the user interface to be closed. The pnc10 has the SRF send a 'Disconnect' message to the user interface and makes the modem at the SRF side hang-up. The dfc10 SIB disconnects the SRF from the user interface. The quit10 SIB ends the script and the control over the call is returned to the SSF.

Whenever the A party is disconnected, the monitor10 SIB triggers the get_calldata10 SIB, which gets the charging data from the call data in the SSP. The store_charging10 SIB stores the data in the SDF.

In the figure only one compare10 SIB is shown, but in reality six of these SIBs determine which command is issued by the user interface. This is because a compare10 SIB is only capable of selecting a *true* or *false* outlet, indicating whether or not a certain command is received. To simplify the figure, this detail has not been depicted.
The second script on the SCF implements the Abbreviated Dialing service. It actually is just a part of a user's script. To enable a certain user to use the Abbreviated Dialing service the sub­script shown in Figure 7-14 must be part of the total script of that user.

The sub-script is started when the user dials 222 followed by two digits which represents the abbreviated terminating number. The monitor10 SIB sets a monitor on the disconnect state. In the modify10 SIB, the prefix 222 is removed from the dialed number, so only the abbreviated number remains. The abbr_dialing10 SIB gets a list of original numbers and a list of abbreviated numbers from the SDF and determines the original number that must be called. If the proper number is found in the SDF, the loaddata10 SIB takes care of setting up a call to the appropriate destination. If not, a message is shown on the terminal’s display to indicate to the user that an error occurred (e.g. the abbreviation is unknown).

When the A party is disconnected, the monitor10 SIB triggers the get_calldata10 SIB, which gets the charging data from the SSF. Finally, the store_charging10 SIB stores the charging data in the SDF.

Figure 7-14: Abbreviated Dialing script
7.8.2.2 SRF Script

When on the SCF the pnc10 SIB is executed, the script that is shown in Figure 7-15 will be started on the SRF.

![Diagram of SRF script]

*Figure 7-15: SRF script*

The main part of the script is the pnc40 SIB which has been described in Table 7-6. When the result of the SIB is *ok* or *nok*, the sendresult40 SIB sends the result to the SCF. If the result was *ok*, the done40 SIB is activated, which causes the script interpreter to wait for an event. When a *stop* or *result* event occurs, the pnc40 SIB is re-entered and the result is collected and stored. Subsequently, the send_ci40 SIB is executed, which sends the collected information to the SCF. If requested the disc40 SIB disconnects the SRF from the SSF and finally the execution of the script ends with the quit40 SIB, and the SCF regains control again.

7.9 Summary and Conclusions

In this chapter the implementation of the IN/CC environment and some IN/CC services have been considered. For the PC side of the system, a user interface has been developed from which the user can manage his user profile, and initiate communication sessions between the user interface and the IN. At the IN side new SIBs and scripts have been implemented to enable these communication sessions. Furthermore, new message have been introduced, enriching the communication protocols. To demonstrate the possibilities some IN/CC services have been implemented in this environment: one profile service (Abbreviated Dialing) and two secure services (Account Card Calling and Universal Personal Telecommunications).

The designed environment provides a good means to implement and test new IN/CC service ideas.
8. Conclusions and Recommendations

This report describes the results of a graduation project, which focused on the use of chipcards in Intelligent Network services. This final chapter presents the conclusions of this report and some recommendations for the future.

8.1 Conclusions

In this paragraph the discussed topics of this report will be shortly reviewed. Where relevant, some conclusions will be drawn. We will start with the four main topics of this report:

- **Chipcard Technology** - In this report a lot of attention was paid to the ISO 7816 standard for chipcards. This standard is the basis for most existing chipcards. The standard however does not provide a complete description that covers all details. This makes chip manufactures flexible in designing chipcards, but it introduces non-compatibility among chipcards from different manufactures. Two important standards exist which can be seen as an extension of the ISO 7816 standard, namely the ETSI TE9 and the EMV standard. Unfortunately, this causes a split in the chipcard technology. The ETSI standard is the most popular standard for telecommunication companies, while the EMV (Eurocard, Mastercard, Visa) is most popular in the financial world. To achieve a situation in which all chipcards can be used from all terminals, an open and worldwide accepted standard is necessary.

- **Charging** - Some charging related issues have been discussed briefly in this report. Four different payment methods have been described:
  - Post-payment:
    - Terminating point dependent
    - Card dependent
  - Pre-payment:
    - Single transaction
    - Sliced transaction

In the first method, no chipcard is involved. The user is charged dependent on the location from where he makes a call. The other three methods use a chipcard. The card dependent post-payment method determines the identity of the user by reading some personal data from the card. So not the owner of the used telephone, but the owner of the chipcard will be charged. The latter two methods withdraw electronic money from a chipcard, in a single transaction respectively in (more than one) sliced transactions. Again the owner of the chipcard pays for the call, rather than the owner of the telephone.

- **Identification and authentication** - Chipcards make reliable identification and authentication of users possible. Identification can easily be done by reading a user ID from the card, which unambiguously determines who owns the card. The user only has to enter a PIN to prove that it is his card. Further, thanks to the chipcard’s capability to perform cryptographic calculations, the involved parties in a communication process can authenticate each other, to prove they are all reliable. The most important cryptographic algorithms to do so, are DES and RSA.
Profile - In this report the standardized IN services (from CS-1 and CS-2) have been analyzed, to determine which services can be extended to IN/CC services. All services which satisfy certain criteria have been selected. It seemed that the selected services can be divided into two groups of IN/CC services:
  - Secure services
  - Profile services

The secure services make use of the just described reliable identification and authentication methods. The profile services make use of the storage space which chipcards provide. A complete service or user profile could be stored on a chipcard to describe the user's personal service parameters. In this way, the user can use his services with his parameters in any operators network. When somebody for instance wants to use his services abroad, he can just insert his chipcard in a terminal and upload his profile to the IN. From that moment his services are available. Further, the information on the card can be changed using some profile management software. For profile management, a good user interface on a PC might be much more user friendly than a voice responding telephone system.

To be able to implement some IN/CC services, an IN/CC prototyping environment has been developed. This environment mainly makes communication between a chipcard and the IN possible. Identification of the user by means of a chipcard and profile management is supported. Authentication of the involved entities has not been implemented, because the used chipcard technology does not support cryptographic calculations.

To demonstrate the use of chipcards in IN services, three different IN/CC services have been implemented:

- Abbreviated Dialing service
- Account Card Calling service
- Universal Personal Telecommunications service

The IN/CC environment has been setup as flexible as possible to allow easy implementation of new service ideas in the future.

8.2 Recommendations

This paragraph will conclude this report by providing some recommendations:

- As discussed in the previous paragraph, compatibility among different chipcards and terminals is not a fact yet. Fortunately, there are parties putting a lot of effort in developing an open system for chipcards. In this context the Java Card API must be mentioned, which makes chipcard development using standard Java tools possible. This open standard is widely supported by many leading chipcard developers and manufactures all over the world. This may result in Java Card becoming the solution for compatibility among chipcards and terminals. Right now it seems to be important to at least closely follow the development of Java Card.
In order to make the use of user profiles on chipcards possible in the future, a lot of effort must be invested in standardization of IN/CC services. Communication protocols between the chipcard side of the system and the IN must be standardized. For instance, the Q.1200 series do not cover the transfer of strings between the user side (e.g. a modem) and the SRF. Further, the structure of the service profiles must be standardized for all IN/CC services, so that the profiles can be used in any Intelligent Network.

This report concentrated on the use of chipcards in *existing* IN services. However, thanks to chipcards many new services could be designed, which are specialized in the use of chipcards. Remote ticketing is a good example. It allows users to call from a telephone with an integrated chipcard terminal and to order, pay and receive tickets (e.g. for the cinema, train, airplane, etc.). The appropriate amount of money is withdrawn from the chipcard’s electronic purse (pre-payment) and an electronic ticket is stored on the card. This is just one example of new IN/CC service, but it will be clear that the introduction of chipcards in IN will greatly expand the possibilities of Intelligent Network services.
9. Literature

[ARM93]  
Concurrent Programming in Erlang  
J. Armstrong, R. Virding, M. Williams  
Prentice Hall International (UK), Ltd., 1993, Hertfordshire  
ISBN 0-13-285792-8

[ACISTD]  
ACI Limited  
Publication Number: BMSP-PL215-01  
August 1994

[CKTS]  
Chipknip Terminal Specification  
BeaNet B.V.  
Release 1.3, September 1996

[CPTS]  
Chipper Payment Terminal Specifications  
Chipper  
Version 1.1, March 1997

[EMV1]  
EMV'96 - Integrated Circuit Card Specification for Payment Systems  
Part 1: Card Specifications  
Europay International S.A., MasterCard International Inc., Visa International S.A.  
Version 3.0, June 30, 1996

[EMV2]  
EMV'96 - Integrated Circuit Card Specification for Payment Systems  
Part 2: Terminal Specifications  
Europay International S.A., MasterCard International Inc., Visa International S.A.  
Version 3.0, June 30, 1996

[EMV3]  
EMV'96 - Integrated Circuit Card Specification for Payment Systems  
Part 3: Application Specifications  
Europay International S.A., MasterCard International Inc., Visa International S.A.  
Version 3.0, June 30, 1996

[HELV97]  
Chipknip and Chipper revealed  
L.D.C. van Helvoort  
Version 1.0, November 1997
[ISO1177]  
*Information processing - Character structure for start/stop and synchronous character oriented transmission*  
International Organization for Standardization  
1985

[ISO7816-1]  
*Identification cards - Integrated circuit(s) cards with contacts*  
*Part 1: Physical characteristics*  
International Organization for Standardization  
First edition, July 1, 1987

[ISO7816-2]  
*Identification cards - Integrated circuit(s) cards with contacts*  
*Part 2: Dimensions and locations of the contacts*  
International Organization for Standardization  
First edition, May 15, 1988

[ISO7816-3]  
*Identification cards - Integrated circuit(s) cards with contacts*  
*Part 3: Electronic signals and transmission protocols*  
International Organization for Standardization  
First edition, September 15, 1989

[ISO7816-3A1] - Amendment 1  
*Identification cards - Integrated circuit(s) cards with contacts*  
*Part 3: Electronic signals and transmission protocols - Amendment 1: Protocol type T=1, asynchronous half duplex block transmission protocol*  
International Organization for Standardization  
First edition, September 15, 1989

[ISO7816-3A2] - Amendment 2  
*Identification cards - Integrated circuit(s) cards with contacts*  
*Part 3: Electronic signals and transmission protocols - Amendment 2: Revision of protocol type selection*  
International Organization for Standardization  
First edition, September 15, 1989

[ISO7816-4]  
*Identification cards - Integrated circuit(s) cards with contacts*  
*Part 4: Interindustry commands for interchange*  
International Organization for Standardization  
First edition, September 1, 1995

[ISO7816-5]  
*Identification cards - Integrated circuit(s) cards with contacts*  
*Part 5: Numbering system and registration procedure for application identifiers*  
International Organization for Standardization  
First edition, May 15, 1994
[ISO7816-6]  
*Identification cards - Integrated circuit(s) cards with contacts*  
*Part 6: Interindustry data elements*  
International Organization for Standardization  
First edition, May 15, 1996  

[KRUG96]  
*Inside Visual C++, the standard reference for programming with Microsoft Visual C++ 4.0*  
D.J. Kruglinski  
Microsoft Press, 1996, Redmont, Washington 98052-6399  
ISBN 1-55615-891-2  

[MART94]  
*TCP/IP Networking*  
J. Martin, J. Leben  

[MFKS]  
*Bestuuringsstructuur voor de Multifunctionele Chipknip*  
Easychip B.V.  
Version 1.0, March 1, 1997  

[OIC97]  
*Nederlandse praktijkrichtlijn Open Infrastructuur voor Chipkaarttoepassingen, NPR 7402*  
Stichting Nationaal Chipcard Platform (NCP)  
First edition, April 1997  
Nederlands Normalisatie-Instituut (NNI), Delft  

[prEN728-1]  
*Identification card systems - Telecommunications integrated circuit(s) cards and terminals*  
*Part 1: System overview*  
European Committee for Standardization  
December 1994  

[prEN728-2]  
*Identification card systems - Telecommunications integrated circuit(s) cards and terminals*  
*Part 2: Security framework*  
European Committee for Standardization  
November 1995  

[prEN728-3]  
*Identification card systems - Telecommunications integrated circuit(s) cards and terminals*  
*Part 3: Application independent card requirements*  
European Committee for Standardization  
December 1994
[prEN728-4]
Identification card systems - Telecommunications integrated circuit(s) cards and terminals
Part 4: Application independent card related terminal requirements
European Committee for Standardization
December 1994

[prEN728-5]
Identification card systems - Telecommunications integrated circuit(s) cards and terminals
Part 5: Payment methods
European Committee for Standardization
January 1994

[prEN728-6]
Identification card systems - Telecommunications integrated circuit(s) cards and terminals
Part 6: Telecommunication features
European Committee for Standardization
November 1995

[prEN728-7]
Identification card systems - Telecommunications integrated circuit(s) cards and terminals
Part 7: Security module
European Committee for Standardization
May 1994

[Q.1200]
Q-Series Intelligent Network Recommendation Structure
International Telecommunication Union
1992

[Q.1202]
Intelligent Network Service Plane Architecture
International Telecommunication Union
1992

[Q.1203]
Intelligent Network Global Functional Plane Architecture
International Telecommunication Union
1992

[Q.1204]
Intelligent Network Distributed Functional Plane
International Telecommunication Union
1992

[Q.1205]
Intelligent Network Physical Plane Architecture
International Telecommunication Union
1992
[Q.1210]
Q-Series Intelligent Network Recommendation Structure
International Telecommunication Union
October 1995

[Introduction to Intelligent Network Capability Set-1]
International Telecommunication Union
March 1993

[Q.1213]
Global Functional Plane for Intelligent Network CS-1
International Telecommunication Union
October 1995

[Q.1214]
Distributed Functional Plane for Intelligent Network CS-1
International Telecommunication Union
October 1995

[Q.1215]
Physical Plane for Intelligent Network CS-1
International Telecommunication Union
October 1995

[Q.1218]
Interface Recommendation for Intelligent Network CS-1
International Telecommunication Union
October 1995

[Q.1220]
Q-Series Intelligent Network Recommendation Structure
International Telecommunication Union
Draft version, 1996

[Q.1221]
Introduction to Intelligent Network Capability Set-2
International Telecommunication Union
Draft version, 1996

[Q.1223]
Global Functional Plane for Intelligent Network CS-2
International Telecommunication Union
Draft version, 1996

[Q.1228]
Interface Recommendation for Intelligent Network CS-2
International Telecommunication Union
Draft version, 1996
[SPDH-E]
InfoBase STANDARD POS DEVICE SUPPORT ENHANCED MANUAL
ACI Limited
Publication Number: BMPB-DH740-51

[SPEC97]
IEEE Spectrum
The Institute of Electrical and Electronics Engineers, Inc.
ISSN 0018-9235
February 1997

[TAN96]
Tanenbaum, Andrew S.
Computer Networks
3rd Edition

[THO94]
Thöner, J.
Intelligent Networks

[V.32bis]
SERIES V: DATA COMMUNICATION OVER THE TELEPHONE NETWORK
A duplex modem operation at data signalling rates of up to 14400 bits for use on the general
switched telephone network and on leased point-to-point 2-wire telephone-type circuit
ITU-T, 1991

[V.42]
SERIES V: DATA COMMUNICATION OVER THE TELEPHONE NETWORK
Error-correcting procedures for DCEs using asynchronous-to-synchronous conversion
ITU-T, 1996

[X.25]
Interface between Data Terminal Equipment (DTE) and Data Circuit-terminating Equipment
(DCE) for terminals operating in the packet mode and connected to public data networks by
dedicated circuit
International Telecommunication Union
October 1996
Appendix A: JPEG Compression

The purpose of this appendix is to give a short introduction to JPEG (Joint Photographic Experts Group), which is a standardized compression technique for images. It has been designed to compress 'continuous' images, like photographs. It is not suitable for line drawings, simple cartoons, black and white pictures, etc. Images with many abrupt transitions from one color to a complete different color will not be compressed well. However, if the color transitions are rather smooth, JPEG is the most favorable compression technique for both full-color and gray-scale images.

Opposed to other compression algorithms (like GIF), JPEG is not lossless, that is, the quality of the image will be reduced to obtain a smaller file. This decrease in quality will be noticed if the original and the resulting image are compared on a pixel by pixel basis, but mostly the human eye will not be able to perceive differences between the two images when looked at as a whole. In this way, JPEG can typically achieve compression rates of about 20:1 without visible loss, compared to a 4:1 rate for GIF compression.

When converting an original image to a JPEG image, two trade offs can be made, namely:

- **Quality versus size** - By decreasing the quality of the image, the size of the resulting file will be reduced. Of course, the decrease of quality will become visible beneath a certain threshold.

- **Quality versus speed** - By decreasing the quality of the image, the encoding and decoding processes will become faster by introducing inaccurate approximations to the required calculations.

By reducing the quality of the image, a compression rate of about 100:1 even is feasible.

Besides the large compression rate, the JPEG algorithm has another advantage over, for instance, the GIF compression algorithm. This is related to the number of colors in an image. When a GIF image is created, the number of colors is determined by the creator of the image and is fixed from that moment. In the past, most computer displays only supported 8-bit images, that is, images with a maximum of 256 colors. Therefore most GIF images only have 256 or less different colors. However, nowadays many displays support 24-bit images with a maximum of over 16 million different colors (which is referred to as full-color). Showing a 8-bit GIF image on such a screen actually is a waste of display capability. On the other hand, showing a GIF image with over 256 colors on a 8-bit screen requires a color reduction, which decreases the quality even more than when the number of colors is reduced once from the original full-color image.
JPEG however is a full-color format, so every JPEG image can contain more than 16 million colors. The advantage of JPEG is, that the reduction of the number of colors is only done when the image is displayed instead of when the image is created (the reduction of the number of colors is called color quantization). The number of colors displayed is of course still dependent on the kind of display being used, but the display capabilities are always completely utilized. If for instance an 8-bit display is used, color quantization maps the required colors as good as possible to the available 256 colors. But the same image will be shown with 16 million colors on a 24-bit display.
Appendix B: Store_charging10 SIB Erlang Code

-module(store_charging10).
-export([initial/2]).

initial(Logicdata,RunList) ->
    io:format("-n-nSTORE_CHARGING10...-n-n"),
    [UserID,Charging_data] = bbsupport10:get_logic_params([user_id,
      charging_data],Logicdata),
    SDF_data=sdf10:get(UserID,user_data,acc_charged),
    SDF_data2=window9(SDF_data,[]),
    Result=append_charging(Charging_data,SDF_data2),
    case Result of
      error ->
        io:format("-n-nCharging data format error!-n-n"),
        [nok,RunList];
      Other ->
        sdf10:put(UserID,user_data,acc_charged,[system,Result]),
        [ok,RunList]
    end.

window9([],SDF_data_new) ->
    SDF_data_new;
window9([H|T],SDF_data_new) ->
    case H of
      [Name, [Type,Value]] ->
        L=length(Value),
        if
          L==9 ->
            Value2=lists:nthtail(L-8,Value),
            window9(T,lists:append(SDF_data_new,
              {[Name, [Type,Value2]]}));
        L<9 ->
            window9(T,lists:append(SDF_data_new, [H]))
        end;
      Other ->
        io:format("-n-nCharging data format error!-n-n")
    end.

append_charging([],SDF_data) ->
    SDF_data;
append_charging([H|T],SDF_data) ->
    case H of
      [Tagname, [Tagtype,Tagvalue]] ->
        Match=lists:keysearch(Tagname,1,SDF_data),
        case Match of
          false ->
            error;
          [value,[Name,[Type,Value]]] ->
            SDF_data2=lists:keyreplace(Tagname,1,
              SDF_data,[Tagname,[Tagtype,
                lists:append(Value,[Tagvalue])]])
            append_charging(T,SDF_data2);
          Other ->
            error
        end;
      Other ->
        error
    end.