MASTER

Facsimile group 4 : research on the compatibility between the 1984 and 1988 CCITT Recommendations, and derivation of conformance tests to test this compatibility

Beckers, H.J.H.

Award date:
1989

Faculty of Electrical Engineering
Eindhoven University of Technology
Department of Telecommunications

FACSIMILE GROUP 4
Research on the compatibility between the 1984 and 1988 CCITT Recommendations, and derivation of conformance tests to test this compatibility.

By: H.J.H. Beckers.

Graduation Report of the final project
carried out from April until December 1989.
Professor: prof. ir. J. de Stigter.
Coach: ir. R.J. Helwerda

Place: Leidschendam.
Date: December 1, 1989.

Abstract

During the CCITT study period from 1985 until 1988 new recommendations for facsimile group 4 were developed. According to CCITT these new recommendations are compatible with the old recommendations, i.e. from the study period 1981 until 1984. Since the structures of these sets of recommendations are completely different, the statement of CCITT is not very plausible. In this report the two sets of recommendations are discussed and checked on their compatibility. The emphasis of this research falls on the top layer of the OSI Reference Model, i.e. the application-layer.

It appears that there are some subtle differences which may cause incompatibilities. These differences vary from the use of the underlying layer, which is in this case the session layer, to the use and coding of several document attributes. So, all in all it can be concluded that CCITT has tried to keep the 1988 recommendations compatible with the 1984 recommendations, although at some points it has failed to do so.

To test whether an implementation has this compatibility, conformance tests were derived. This was done according to the ISO Conformance Testing Methodology and Framework standard. Instead of the "Tree and Tabular Combined Notation", which is specified in the standard mentioned, another notation is used. This notation is developed by the Research Neher Laboratory, and consists of a subset of the notation mentioned above.

The main problem that occurred, was the fact that the test cases are not derived to test conformance, since interworking is tested rather than conformance. This makes the assignment of a verdict, according to the Conformance Testing Methodology and Framework standard, to indicate compatibility, not possible. As a solution to this problem a variable is suggested, to indicate whether or not an implementation is compatible. The value of this variable is, via an extra point of control and observation, saved at a predefined memory location in the protocol analyzer. This makes the testresults accessible for the tester.

Samenvatting

Tijdens de CCITT studie periode van 1985 tot en met 1988 zijn nieuwe standaarden voor facsimile groep 4 ontwikkeld. Volgens CCITT zijn deze nieuwe standaarden compatibel met de oude standaarden, d.w.z uit de studie periode van 1981 tot en met 1984. Aangezien de strukturen van beide sets van standaarden dermate sterk van elkaar verschillen, is de bewering van CCITT niet erg geloofwaardig. In dit afstudeerverslag worden de beide standaarden besproken en onderzocht op hun compatibiliteit. Hierbij ligt de nadruk op de bovenste laag van het OSI Referentie Model, m.a.w. de applicatie-laag.

Het blijkt dat er enkele subtiele verschillen bestaan, die kunnen leiden tot problemen met betrekking tot de compatibiliteit. Deze verschillen variëren in het gebruik van de onderliggende laag, hetgeen in dit geval de sessie-laag is, tot het gebruik en de codering van enkele document attributen. Al met al kan gesteld worden dat CCITT geprobeerd heeft om de 1988 standaarden compatibel te houden met de 1984 standaarden, hetgeen echter op sommige punten niet gelukt is.

Om nu te kunnen testen of een bepaalde implementatie deze compatibiliteit ook daadwerkelijk bezit, zijn conformiteits tests afgeleid. Deze afleiding is volgens de ISO standaard "Conformance Testing Methodology and Framework" uitgevoerd. Hierbij is echter i.p.v de "Tree and Tabular Combined Notation" een afwijkende notatie voor de testreeksen gebruikt. De gebruikte notatie is door het Research Neher Laboratorium ontwikkeld, en bestaat uit een subset van bovengenoemde notatie.

Het belangrijkste probleem dat hierbij optrad, bestond uit het feit dat deze testreeksen niet afgeleid werden om conformiteit te testen, maar om "interworking" te testen. Om nu het resultaat van een test te kunnen rapporteren, kan men niet gebruik maken van "verdict" toekenning. Als oplossing voor dit probleem is gekozen voor een variabele die aangeeft of een implementatie compatibel is of niet. De waarde van deze variabele wordt via een extra "point of control and observation" op een vooraf vastgestelde geheugen locatie in de protocol analyser bewaard. Hierdoor wordt uitlezing van het testresultaat mogelijk.

Contents

List of abbreviations

ACSE	Association Control Service Element
ALP	Abstract Local Primitive
APDU	Application Protocol Data Unit
ASN.1	Abstract Syntax Notation 1
ASAP	Application Service Access Point
ASP	Abstract Service Primitive
BMU	Basic Measurement Unit
CCITT	Comité Consultative International de Télégraphique et Téléphonique
CCR	Commitment Concurrency Recovery
CDC	Command Document Continue
CDCL	Command Document Capability List
CDE	Command Document End
CDP	Command Document Page Boundary
CDS	Command Document Start
CDUI	Command Document User Information
CSPDN	Circuit Switched Public Data Network
CSS	Command Session Start
CSUI	Command Session User Information
CTS	Conformance Testing Services
DATAM	Document Architecture Transfer And Manipulation
DCR	Dynamic Conformance Requirements
DTAM	Document Transfer And Manipulation
EC	European Community
EOFB	End Of Facsimile Block
ETSI	European Telecommunication and Standardization Institute
FAX 4	FACSimile Group 4
FDA	Formatted Document Architecture
FPDA	Formatted Processable Document Architecture
FTAM	File Transfer Access and Management
HDLC	High level Data Link Control
ISDN	Integrated Services Digital Networks
ISO	International Organization for Standardization
IUT	Implementation Under Test

LAN	Local Area Network
LT	Lower Tester
MHS	Message Handling Systems
MRC II	Modified Read Code II
ODA	Open Document Architecture
ODIF	Open Document Interchange Format
OSI	Open Systems Interconnection
PAD	Packet Assembler Disassembler
PCO	Point of Control and Observation
PCTR	Protocol Conformance Test Report
PDN	Public Data Network
PDA	Processable Document Architecture
PDH	Horizontal Dimension of Page
PDV	Vertical Dimension of Page
pel	picture element
PICS	Protocol Implementation Conformance Statement
PIXIT	Protocol Implementation eXtra Information for Testing
ppi	pels per inch
PSPDN	Packet Switched Public Data Network
PSTN	Public Switching Telephone Network
RDCLP	Response Document Capability List Positive
RNL	Research Neher Labs
ROSE	Remote Operation Service Element
RTSE	Reliable Transfer Service Element
SASE	Specific Application Service Element
SAP	Service Access Point
SCR	Static Conformance Requirements
SCTR	System Conformance Test Report
SLP	Synchronous Link Protocol
SMU	Scaled Measurement Unit
SPDU	Session Protocol Data Unit
SSAP	Session Service Access Point
SUD	Session User Data
SUT	System Under Test
TA	Terminal Adapter
TCP	Test Coordination Procedures
TIF	Text Image Format
TMP	Test Management Protocol
TPF	Text Processable Format
TSAP	Transport Service Access Point
TTCN	Tree and Tabular Combined Notation
UT	Upper Tester
WAN	Wide Area Network

Chapter 1

Introduction

This report is a graduation report of the final project, done at the Faculty of Electrical Engineering of the Eindhoven University of Technology. Since the project was an assignment of the Research Neher Laboratory (RNL) of the Dutch PTT, all the work was done at this company.

The final project aimed at the investigation of the compatibility between the 1984 CCITT[1] Recommendations and the 1988 CCITT Recommendations for facsimile group 4. In case of incompatibilities, conformance tests, aimed at those functionalities where problems of compatibility can occur, should be derived. According to CCITT the 1984 and 1988 recommendations are compatible. The RNL however has its doubts about this, because the structures of both sets of recommendations are completely different. This makes the statement of CCITT difficult to verify and not very plausible. In fact, the recommendations show subtle incompatibilities which will be discussed in this report.

The project mentioned above was part of a bigger project of RNL, the Conformance Test ISDN[2] B-channel protocol project. This project was done within the scope of the CTS[3]-2-ISDN project of the European Community. These projects and the need of conformance testing are discussed in chapter 2.

In chapter 3 a closer is taken at the 1984 recommendations for facsimile group 4, and the same will be done for the 1988 recommendations, in chapter 4.

Incompatibilities and vagueness between these two standards will be the subject of chapter 5.

---

[1]CCITT = Comité Consultative International de Télégraphique et Téléphonique

[2]ISDN = Integrated Services Digital Network

[3]CTS = Conformance Testing Services

Chapter 6 discusses the theory of conformance testing, which is standardized in OSI[4] Conformance Testing Methodology and Framework (ISO[5] 9646) part 1 up to 5. Of these five parts only the first three parts are of interest for this project.

In chapter 7 this theory is applied to derive conformance tests for the purpose mentioned above. Furthermore the test method used will be discussed and the test cases derived will be given.

Finally, in chapter 8 the conclusions of this research will be given and the work that still has to be done is mentioned.

Many people were willing to give clarifications and explanations when I asked them. There are to many to enumerate them all, but I wish to thank explicitly my coach ir. R. J. Helwerda, drs. E. M. Dijkerman, ir. J. C. Niestern, ir. Y. Yang, ir. J. Koomen, ir. A. M. Peeters, J. Scheffel, drs. A. G. Kleppe, ir. P. H. C. A. van den Merkhof and A. van Halderen. As last I want to thank my professor ir. J. de Stigter for his work and the constructive criticism he had.

[4]OSI = Open Systems Interconnection
[5]ISO = International Organization for Standardization

Chapter 2

The CTS-B project

2.1 The importance of conformance testing

The field of conformance testing of communication protocols is becoming more and more important, and at the moment attracts a great deal of international attention. As a result of the growing use of data-communication and the appearance of different implementations of different protocols on the market, the determination of the conformance of an implementation to a standard becomes very important. The differences in implementations are caused by ambiguities in the standards resulting in different manners of interpretation of these standards.

ISO is an international organization, which has as one of its tasks the standardization of OSI protocols. Deriving test suites (i.e. a set of test cases to test the conformance of an implementation to a standard) could be done by every company working in the field of OSI protocols. However, since these protocols are standardized by ISO, it will be clear that standardization of test suites must also be a task for ISO. By doing so, it will be possible to attach an internationally accepted certificate to an implementation conforming to the relevant standard. These test suites can be used by suppliers or implementors in self-testing, by users of OSI products, by telecommunication administrations and recognized private operating agencies or by other third party testing organizations.

From a vendor point of view, conformance testing is important because:

- it reduces pairwise testing, i.e. testing of interoperatibility with implementations of other manufacturers is not required;
- it makes equipment more marketable;
- only one test is needed;

- testing can be done everywhere, because test results will be the same.

Now we have stated why and for whom conformance testing is important, we will discuss what conformance testing implies. Within ISO the standard OSI conformance testing methodology and framework (ISO 9646, [19]) is developed. This standard defines conformance testing as:

"Conformance testing involves testing both the capabilities and behaviour of an implementation, and checking what is observed against both the conformance requirements in the relevant standard(s) or recommendation(s) and what the implementor states the implementation's capabilities are."

When a conformance test shows that two implementations of the same protocol conform to the standard, this does not mean that these implementations can communicate faultless. The probability however that these different implementations are able to interwork is increased.

Another problem that arises is the fact that the complexity of most protocols makes exhaustive testing impractical on both technical and economic grounds.

2.2 The CTS-2-ISDN project

The first attempts to achieve generally acceptable testing for OSI products were started in may 1986. At that time the CTS-LAN[1] and CTS-WAN[2] programmes emerged as part of an European initiative.

The CTS-2-ISDN project is one of the projects in the second phase off the CTS-WAN program. The goal of the CTS-2-ISDN project is to develop conformance test services for the ISDN D-channel protocols as well as for a group of B-channel protocols (Teletex and Facsimile group 4).

A consortium of companies in countries within the EC[3] is working on this CTS-2-ISDN project. One of these companies is the Dutch PTT. The Research Neher Laboratory of this company will deliver one part of the Dutch contribution to the work to be done in the CTS-2-ISDN project. The other part, the set up of a test laboratory, will be delivered by PTT Contest. At RNL there are two groups working on the CTS-2-ISDN project. These groups are:

- Conformance Test ISDN D-channel project,

[1] LAN = Local Area Network
[2] WAN = Wide Area Network
[3] European Community

- Conformance Test ISDN B-channel project.

Figure 2.1 shows where to put these projects within the Dutch PTT. A dashed line gives the relation between the client and the server.

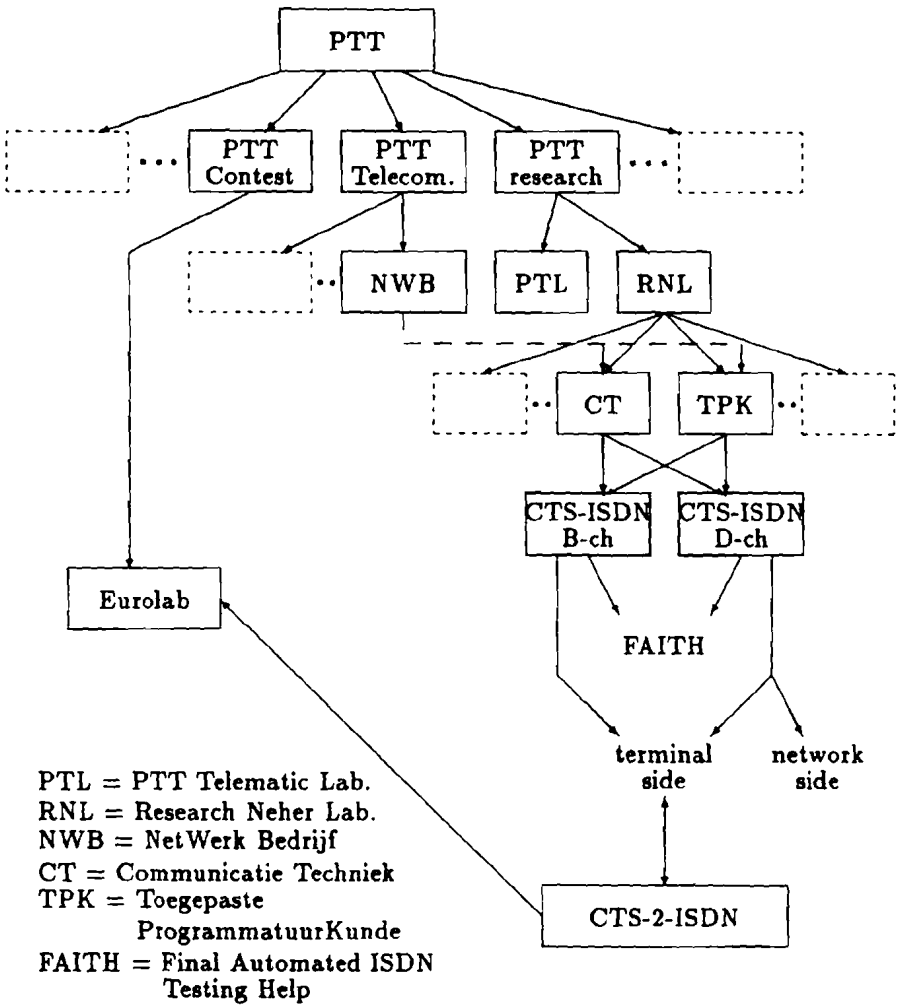


Figure 2.1: *The environment of the CTS-ISDN D-channel and CTS-ISDN B-channel project.*

2.3 The CTS-ISDN B-channel project

The name of this project will be abbreviated to CTS-B project. This project works on conformance tests for teletex and facsimile group 4 services.

The goals of the CTS-B project are summarized below, see [3] and [4]:

1. Development of abstract test suites for B-channel protocols of teletex and facsimile group 4, using the CCITT and ETSI[4] recommendations (see Table 2.1).

	Teletex	Telefax Group 4
layer 7	T.60/T.61	T.503/T.521/T.563/T.6
layer 6	———[1]	———
layer 5	T.62/T.62bis	T.62/T.62bis
layer 4	T.70	T.70
layer 3	ISO8208 and T.70NL	ISO8208 and T.70NL
layer 2	X.75 (SLP)	X.75 (SLP)
layer 1	I.430	I.430

[1] In table 2.1 layer 6 is left empty. Normally the purpose of this layer is to provide a transfer syntax (e.g. Abstract Syntax Notation 1). By doing so, OSI apparatus with different internal data structures can communicate. However, in practice it appears that apparatus with different data structures are not connected. Therefore layer 6 is left empty (transparent).

Table 2.1: *Protocol stack used.*

2. Implementation of the test suites using as much as possible the software tools used for testing the D-channel protocols.
3. Implementation of a reference-implementation.
4. Delivery of the 'deliverables'[5] to the EC within the scope of the CTS-2-ISDN project, and delivery of complete documentation and transfer of knowledge to the operational departments.

The project was started on 15 september 1988 and is planned to finish end of november 1990. The total human effort is estimated on 108 man months.

References

[1], [2], [3], [4], [19], [22].

[4] ETSI = European Telecommunication and Standardization Institute
[5] Reports to the European Community on predefined dates

Chapter 3

Facsimile group 4, the 1984 CCITT Recommendations

3.1 Introduction

This chapter discusses the facsimile group 4 (FAX 4) as defined in the 1984 CCITT T-Series Recommendations[1]. In recommendation T.0 a classification of the facsimile apparatus for document transmission over public networks is given. The document facsimile apparatus can be classified in the following four basic categories:

Group 1: apparatus which use double sideband modulation without any special measures to compress the bandwidth of the transmitted signal.

Group 2: apparatus which exploits bandwidth compression techniques, that includes encoding and/or vestigial sideband working but excludes processing of the document signal to reduce redundancy.

Group 3: apparatus which includes means for reducing the redundant information in the document signal prior to the modulation process (bandwidth compression of the line signal may be incorporated). Most of the FAX terminals in the world are currently of this type.

Group 4: apparatus which incorporates means for reducing the redundant information in the document signal prior to the transmission and assures essentially error-free reception of the document. In contrast with the previous apparatus, which were analog devices, this type of FAX terminals are digital devices. This group is the subject of this report (FAX 4).

[1]The relevant recommendations are: T.0, T.5, T.6, T.62 and T.73

Another difference between the first three groups and the fourth group, is the fact that the former classify apparatus for use over the *public telephone network*, while the latter classifies apparatus for use over the *public data networks* (utilization of an appropriate modulation process makes the use over the public telephone network possible).

To realize mutual communication between the different facsimile groups an Interworking Unit[2], for signals between PDN and PSTN, can be used.

The general aspects of FAX 4 apparatus and their coding schemes and coding control functions are given in the recommendations T.5 and T.6 respectively. In the next section a short discussion of these recommendations is given. The main part of this chapter is dedicated to the document interchange protocol for the telematic services, [5, T.73]. Since this recommendation utilizes recommendation T.62, we will first take a closer look at this recommendation in section 3.3.

The general structure of the teletex/facsimile/mixed-mode common protocols as defined in the 1984 recommendations is given in Figure 3.1 below. Since the study is focussed on the higher layers of the OSI reference model, the transport layer protocol (recommendation T.70) is not discussed.

Figure 3.1 does not show the connection to an ISDN. This can be done with X.21, in which case a terminal adapter (TA) is necessary. A direct connection to the ISDN (at the S or T interface) was still under study in 1984 (see also the CCITT I-Series Recommendations, I.430).

3.2 The general aspects of FAX 4

In this section the general aspects of FAX 4 will be discussed. These general aspects are specified in recommendation T.5. The coding methods used by FAX 4 and specified in recommendation T.6 are discussed in section 3.2.2.

3.2.1 General aspects

As mentioned, FAX 4 terminals are used over the Public Data Networks (PDN). These include the Circuit Switched Public Data Network (CSPDN), the Packet Switched Public Data Network (PSPDN) and the Integrated Services Digital Network (ISDN). The terminal may also be used on the Public Switched Telephone Network (PSTN), using a modem. In all types of network the FAX 4 terminal will

[2]In this case this will be a FAX packet assembly/disassembly (PAD)

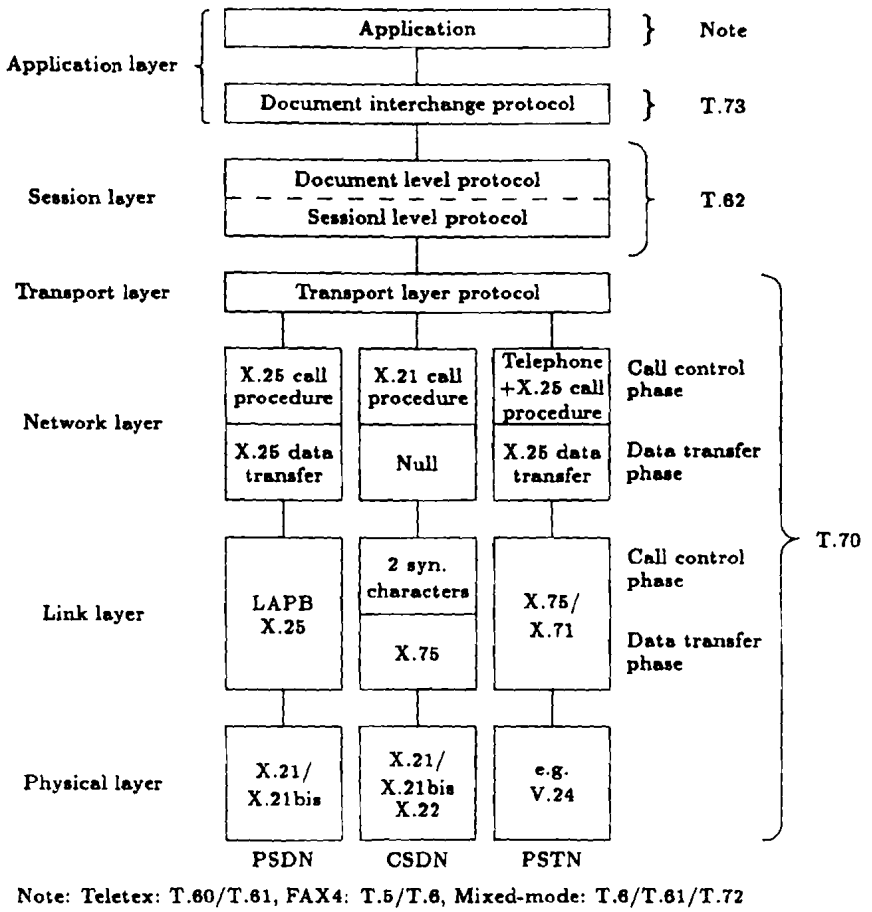


Figure 3.1: *General structure of teletex/facsimile/mixed-mode common protocols*

provide automatic answering, transmission, reception and clearing.

Three classes of FAX 4 terminals can be distinguished:

- Class I :terminal able to send and receive documents containing facsimile encoded information.
- Class II :terminal able to transmit documents which are facsimile encoded, and capable of receiving documents which are facsimile coded, teletex coded and mixed-mode[3].
- Class III:terminal able to generate, transmit and receive facsimile coded, teletex coded and mixed-mode documents.

[3]mixed-mode means that the information content has been encoded using different techniques (facsimile or character coding) and that the document structure has been fully identified, enabling the recipient to apply sophisticated editing methods.

All these classes must provide the following basic characteristics/functions.

- The page (A4/North American) is the base for facsimile message formatting and transmission,
- The content, layout and format of the facsimile message must be identical at the transmitting and receiving terminal,
- The message area should be scanned in the same direction (from left to right) in the transmitter and receiver,
- Each terminal should be equipped with an unique identification,
- The size of the guaranteed reproducible area for ISO A4 (210x297 mm) paper size is 196,6x281,46 mm,
- The requirements for the pel (= picture element) transmission density are given in the Table 3.1. For the paper positioning, center line (e.g. (number of pels/line)/2)) referencing will be used. The raster point in the upper left corner of an ISO page, termed the (1,1) raster reference point, is used as a starting point for determining character margins and positions. To achieve a high service quality, the pel density of the scanner and printer should be greater than or equal to the transmission pel density,

	ISO A4	North American	ISO B4	ISO A3
Resolution (pels/in)				
200	1728/2339[1]	1728/2200	2048/2780	2432/3307
240	2074/2806	2074/2640	2458/3335	2918/3969
300	2592/3508	2592/3300	3072/4169	3648/4961
400	3456/4677	3456/4400	4096/5559	4864/6614
Scan line length (mm)(P)	219,46	219,46	260,10	308,86
Paper width (mm)(Q)	210	215,9	250	297
$P - Q$	9,46	3,56	10,10	11,86
Nominal paper length (mm)	297	279,4	353	420

[1]Number of picture elements along a scan line/Nominal number of scan lines per page

Table 3.1: *Requirements for transmission pel density.*

- The coding scheme used is described in section 3.2.2,
- The use of optional functions can be negotiated during a handshaking procedure in the end–to–end control procedure described in the next section,
- The receiving storage of the terminal is different for the class I terminal which does not require any, and the class II/III terminal, which require a minimum storage of 128 K octets (based on a pel transmission density of 300 pels/inch (ppi) for an ISO A4 document),
- The bit-rates depend on the network, and can be 2.4, 4.8, 9.6 or 48 kbits/s (64 kbits/s on ISDN),

- The frame structure used is the HDLC frame. The apparatus also adopts the HDLC control procedures to enable error free reception by automatic retransmission techniques.

3.2.2 The coding schemes

The facsimile coding schemes consist of the basic facsimile coding scheme (Modified READ Code II (MRC II), for black and white images) and optional facsimile coding schemes (also for grey scale and color images). The use of these optional coding schemes is subject to mutual agreement between the terminals. First the basic facsimile coding scheme, MRC II, is discussed.

The basic facsimile coding scheme and control functions

This coding scheme uses a two-dimensional line-by-line coding method. Every scan line is divided in a number of picture elements. The position of each changing picture element (i.e. an element whose color is different from that of the previous element along the same scan line) on the current coding line is coded with respect to the position of a corresponding reference element. This corresponding element can be situated either on the coding line, or on the reference line which is immediately above the coding line. After the coding line has been coded, it becomes the reference line for the next coding line. The reference line for the first coding line on a page is an imaginary white line.
The position of each changing element along the coding line is coded using one of the following three coding modes:

- pass mode (see Figure 3.2),
- vertical mode (see Figure 3.3),
- horizontal mode (see Figure 3.3).

These modes are coded according to the coding table given in Table 3.2 below.

Now the coding procedure of MRC II is discussed.
First the determination of the changing element b_1 must be explained. This picture element is the first changing picture element to the right of a_0 (the starting picture element), on the reference line, but with opposite color to a_0. b_2 is the first changing picture element after b_1 on the reference line.

How does the coding proceed?
First, detect the various changing picture elements (a_0, a_1, a_2, b_1 and b_2).

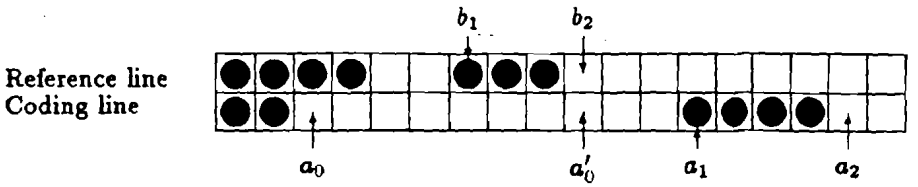


Figure 3.2: *Passmode.*

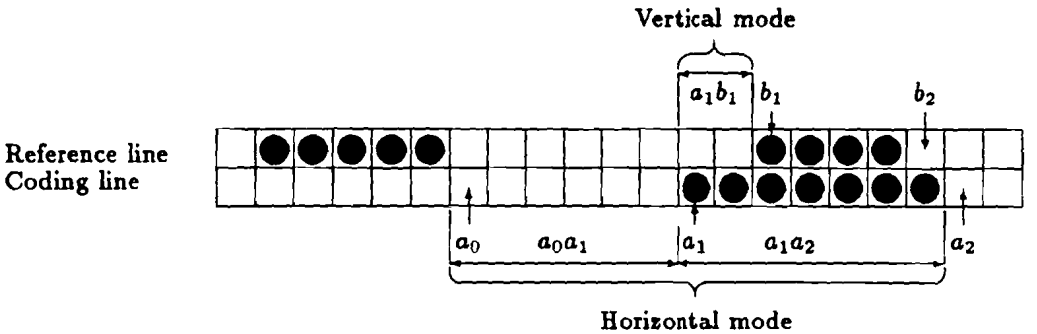


Figure 3.3: *Vertical and horizontal mode.*

Second, determine which mode has to be used.

Pass mode: this mode is used when the position of a_1 is to the right of the position of b_2. In this case the picture element beneath the position of b_2 will become the new position of a_0 after coding (i.e. a'_0). The position of the picture elements a_1 and a_2 do not change (the position of b_1 and b_2 have to be determined again).

Vertical mode: this mode is used when the position of a_1 is to the left of the position of b_2 and when the distance between the position of a_1 and b_1 is smaller than 4 picture elements. The coding depends on this distance (see

Mode	Elements to be coded		Notation	Code word
Pass	b_1,b_2		P	0001
Horizontal	a_0a_1,a_1a_2		H	001 + $M(a_0a_1)$ + $M(a_1a_2)$ [1]
Vertical	a_1 just under b_1	$a_1b_1 = 0$	V(0)	1
	a_1 to the right of b_1	$a_1b_1 = 1$	$V_R(1)$	011
		$a_1b_1 = 2$	$V_R(2)$	000011
		$a_1b_1 = 3$	$V_R(3)$	0000011
	a_1 to the left of b_1	$a_1b_1 = 1$	$V_L(1)$	010
		$a_1b_1 = 2$	$V_L(2)$	000010
		$a_1b_1 = 3$	$V_L(3)$	0000010
Extension				00000011xxx

[1]The codes M() can be found in recommendation T.6.

Table 3.2: *Coding table.*

Table 3.2). After coding, the new position of a_0 will be equal to the position of a_1 before coding. The position of a_1 after coding will be equal to the position of a_2 before coding. The position of a_2, b_1 and b_2 after coding have to be determined again.

Horizontal mode: this mode is used in all the other cases where pass mode or vertical mode are not appropriate. In this case the setting of the changing picture elements (except a_0) after coding have to be determined again. The position of a_0 after coding is equal to the position of a_2 before coding.

It will be clear that, when not taking into account the restrictions on the relative positions between a_1b_1 and a_1b_2 mentioned above, all could be coded using only the horizontal mode. However when coding text, the distance between a_1 and b_1 will generally be very small. This means that vertical mode coding becomes important, because shorter codes are used.
The same can be said about the pass mode. When on the coding line no picture elements of the same color as the picture elements, immediately above the picture elements of the coding line, on the reference line are found, it will not be efficient to use horizontal mode with, for instance, a long white run length.
So the MRC II code is efficient for information consisting of horizontal and vertical lines (e.g. letters and numbers).

The processing of the first and last picture element needs some further explanation. The first starting picture element on each coding line is imaginarily set at a position just before the first actual picture element, and is regarded as a white picture element. The first run length on a line a_0a_1 is replaced by $a_0a_1 - 1$(see Figure 3.4).

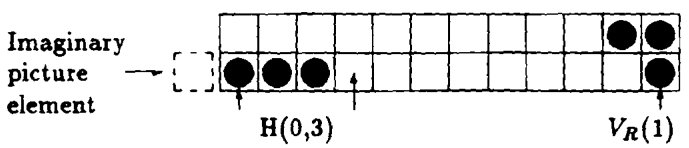


Figure 3.4: *Processing of the first picture element (coding example).*

The coding of the coding line continues until the position of the imaginary changing element situated just after the last actual picture element has been coded. This may be coded as a_1 or as a_2. Also, if b_1 and/or b_2 are not detected at any time during the coding of the line, they are positioned on the imaginary changing element situated just after the last actual picture element on the reference line (see Figure 3.5).

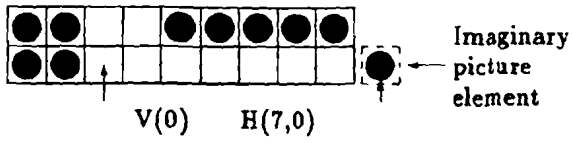


Figure 3.5: *Processing of the last picture element (coding example).*

The *control functions* used are:

- End–of–facsimile-block (EOFB),which is added to the end of every coded facsimile block. The format of EOFB is: 000000000001000000000001.
- Pad bits which are used after the EOFB code to align on octet boundaries or to a fixed block size. The format is: variable length string of 0's.
- Extension which is used to indicate a change from the current coding scheme to another. The format is: 0000001xxx (xxx = 111, i.e. uncompressed).

The optional facsimile coding schemes

The optional facsimile coding schemes for grey scale and color are for further study. For black and white images there is an optional coding scheme: *The uncompressed mode*. This scheme is used to transmit the image information without data compression techniques. The coding table is given in Table 3.3. The image pattern indicates the succession of white (0) and black (1) pels.

Entrance code to uncompressed mode	Basic coding scheme: 0000001111	
	Image pattern	*Code word*
Uncompressed mode code	1	1
	01	01
	001	001
	0001	0001
	00001	00001
	00000	000001
Exit from uncompressed mode code		0000001T[1]
	0	00000001T
	00	000000001T
	000	0000000001T
	0000	00000000001T

[1]T denotes a tag bit which tells the color of the next run (*black* = 1, *white* = 0).

Table 3.3: *Uncompressed mode code words.*

Example:

b	w	w	w	w	w		w	w	b	b	b	w	b	w	→ *compressed*								
1	0	0	0	0	0	1	0	0	1	1	1	0	1	0	0	0	0	0	0	0	1	T

3.3 Control procedures

In this section the session layer of the telematic services will be discussed. From Figure 3.1 it is seen that the session layer protocol is defined by recommendation T.62. Together with the transport layer protocol (recommendation T.70), the session layer protocol is used for reliable transfer of teletex, FAX 4 and mixed-mode documents.

3.3.1 Commands and responses

The session layer protocol consists of a session level protocol and a document level protocol. The document level can be seen above the session level in the OSI model. The relation between the two levels is illustrated in Figure 3.6. The commands and responses provided to implement the control procedures for both level protocols are given in the Table 3.4 and Table 3.5. We will now take a closer look at the commands and responses of these protocols, which are especially interesting for the *document interchange protocol*, as described in section 3.4.

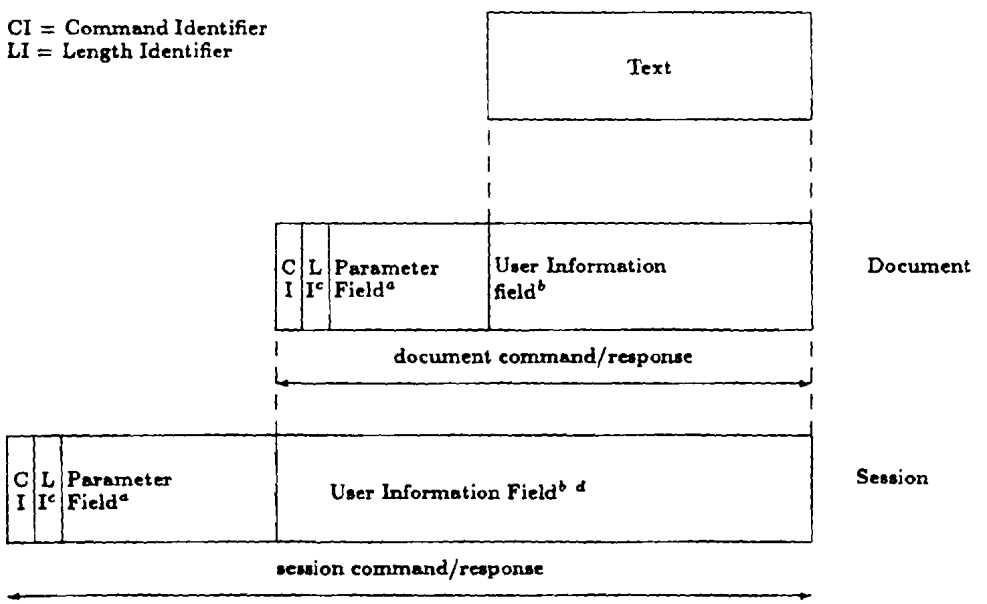


Figure 3.6: *Illustration of the relationship between session and document commands and responses.*

command	response	abbreviation
Session establishment and clearing		
Command session start		CSS
	Response session start positive	RSSP
	Response session start negative	RSSN
Command session end		CSE
	Response session end positive	RSEP
Command session abort		CSA
	Response session abort positive	RSAP
Information transfer		
Command session user information		CSUI
	Response session user information	RSUI
Session management		
Command session change control		CSCC
	Response session change control positive	RSCCP

Table 3.4: *Session commands and responses.*

3.3.2 Session level

This subsection discusses those commands and responses of the session level protocol that are used by the document interchange protocol for negotiation. There are no commands or responses used for invocation. With invocation transport of the document data is meant. The commands and responses of the session level protocol that are important for the document interchange protocol are:

- The *Command session start* (CSS) initiates entry into session. Only the terminal that has established the transport connection (the calling terminal) shall send the CSS. It has fourteen parameters of which only the session user data parameter is important for negotiation in case of FAX 4 application. The session user data parameter is used to convey data of the presentation and/or application protocol.

- The *Response session start positive* (RSSP) is used to acknowledge entry into a session (CSS). This command has twelve parameters. Here also the non-basic terminal capabilities and the session user data are of interest for this study.

Command	Response	Abbreviation
Document control		
Command document start		CDS
Command document continue		CDC
Command document capability list		CDCL
	Response document capability list positive	RDCLP
Command document end		CDE
	Response document end positive	RDEP
Command document discard		CDD
	Response document discard positive	RDDP
Command document resynchronize		CDR
	Response document resynchronize positive	RDRP
Information transfer		
Command document user information		CDUI
Error recovery		
	Response document general reject	RDGR
Command document page boundary		CDPB
	Response document page boundary positive	RDPBP
	Response document page boundary negative	RDPBN

Table 3.5: *Document commands and responses.*

- The *Command session user information* (CSUI) is used to convey commands, parameters and information for the document procedures in the associated information field.

3.3.3 Document level

This subsection discusses the document level protocol commands and responses that are use by the document interchange protocol for negotiation or invocation. These commands and responses are:

- The *Command document start* (CDS) indicates the start of a document. It also indicates the start of the first page. There are six parameters associated with this command. Especially the session user data parameter is of interest, because this parameter is used to convey data of the presentation and/or application protocol(s). This command has no response, except in the case of an error the RDGR command is received. The command is used for *invocation.*
- The *Command document continue* (CDC) indicates to the receiver of this command to continue with the transmission of a document which has been partially transmitted. There are six parameters associated to this command. The session user data parameter has the same function as in the case of the CDS command. Also, like the CDS command, there is no response to the CDC command except the RDGR command in the case of an error. The command is used for *invocation.*
- The *Command document capability list* (CDCL) can only be invoked outside document boundaries, and is used for *negotiation* to:
  - enable a check on the terminal capabilities,
  - investigate the storage capability of the remote terminal,
  - negotiate the value of the inactivity timer,
  - convey the session user data of the presentation and/or application protocol,
  - ascertain compatibility regarding the use of non-standardized capabilities.
- The *Response document capabilities list positive* (RDCLP) acknowledges the CDCL. In case of the terminal capabilities requested, it contains one of the following answers:
  - all requested capabilities are available at the receiver,
  - a list of the requested capabilities that are available,
  - a complete list of non-basic receiving capabilities,
  - no extended capabilities are available in the terminal, or none of the capabilities indicated in the CDCL are available.

  In case of memory negotiation one of the following shall be included:
  - amount of memory requested is available and reserved,
  - available (and reserved) amount of memory,
  - requested memory capacity can not be reserved now,
  - the available memory can not be estimated.

  Because the command is a response to the CDCL command, it is also used for negotiation of the inactivity timer value, to convey session user data of the layers above and to ascertain compatibility regarding the use of non-standardized capabilities.

- The *Command document user information* (CDUI) is used to convey user text information, i.e. the actual document information to be transmitted.

3.3.4 Error recovery

In case of an abnormal termination of a document, the checkpoint reference number and the document reference number are required for recovery in the same session. The checkpoint reference number is, in the basic services, inserted by the *Command document page boundary* (CDPB) and must be explicitly acknowledged. A final checkpoint will be given by the *Command document end* (CDE). The document reference number is inserted by the *Command document start* (CDS). In case of an abnormal termination of a session/call, a new session has to be initiated. To identify the point from which to recover the reference of the interrupted session, the document reference number and the checkpoint reference number are needed.

Note: At the moment the FAX 4 terminals available on the market do not incorporate a possibility to resume after an error. The transmission of a document has to start all over again from the beginning.

3.4 The document interchange protocol

The subject of this section is the document interchange protocol (DIP), which specifies the document architecture and interchange of text. Figure 3.7 shows the relation between the DIP and the other recommendations described.

The way the document is interchanged is called the interchange format. An interchange format is a set of objects and attributes that enables the recipient to reconstruct the received document of a certain complexity. For the next discussion, a document is defined as an amount of text that is interchanged between telematic terminals for two major reasons:

- It may be interchanged as an original in a final form allowing for printing, displaying and storing by the recipient (Text Image Format);
- It may be interchanged in a revisable form, allowing for processing by the recipient (Text Processable Format).

The document architecture contains two structures:

1. The *layout* structure for positioning and rendition on the presentation media.

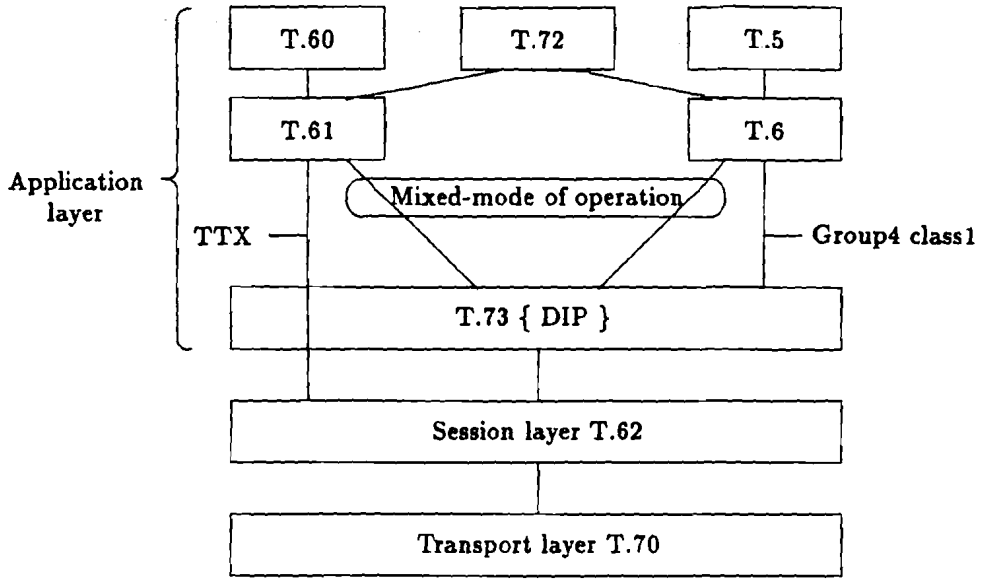


Figure 3.7: *Framework of recommendations for telematic services*

2. The *logical* structure e.g. chapters, sections, headings, paragraphs, etc.

Within these structures a *specific* (i.e. pertaining to a particular document) part and a *generic* (i.e. predefined content portions) part are distinguished. The main advantage of the generic part is that it can speed up the transmission by means of substitution mechanisms.

3.4.1 The structuring of text

A document consists of a document profile and a document body. This document profile is an introductory part and includes information for handling the document as a whole.

The document layout structure consists of the specific layout structure and the generic layout structure. The specific layout structure is a tree structure (see Figure 3.8).

The generic layout structure consists of a set of predefined generic layout objects to which specific layout objects may refer. There are five types of generic layout objects:

- the generic document (root),
- the generic page set,
- the generic page,

- the generic frame and
- the generic block.

These objects form the same tree structure as the specific layout objects.

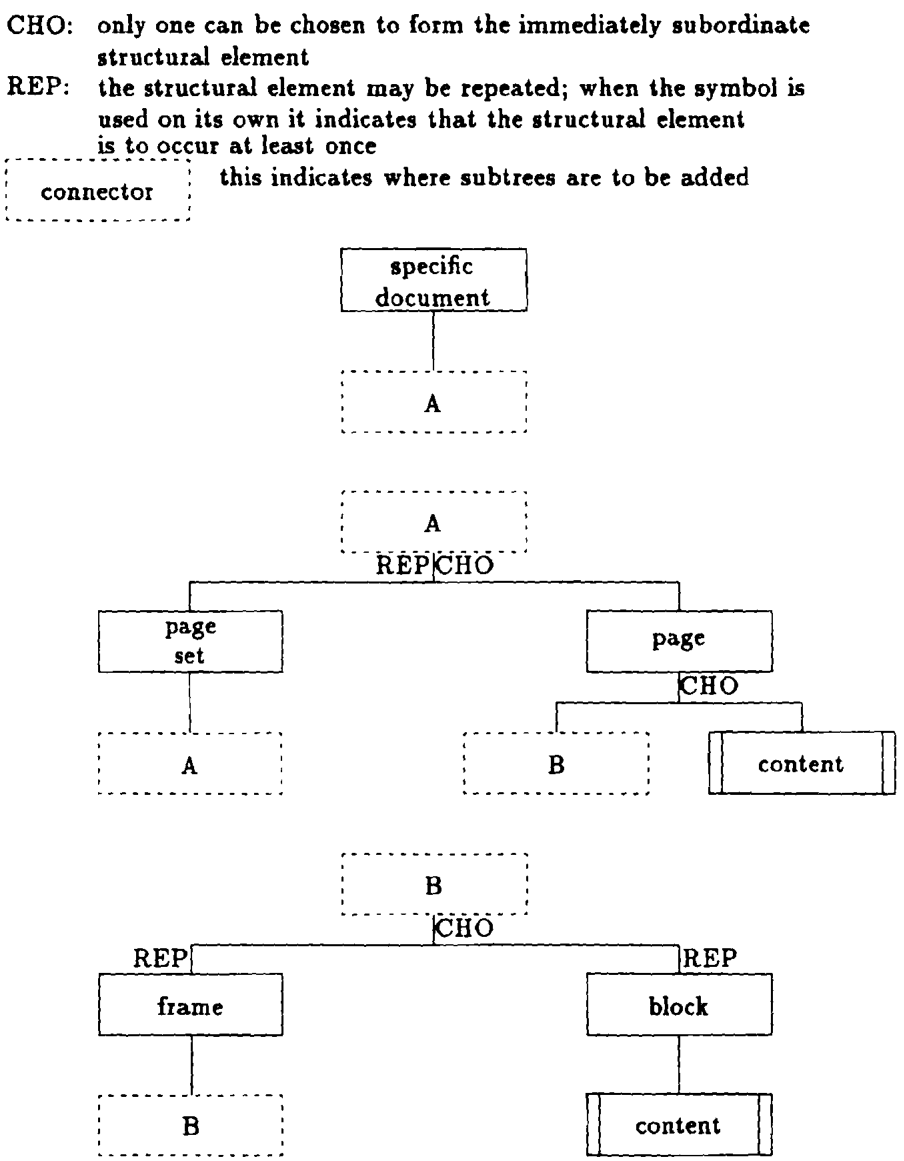


Figure 3.8: *The specific layout structure.*

How is the generic layout structure used?
Whenever a common content portion (e.g. logos, copyright notices, etc) occurs in a specific document, there is a specific block in the specific layout structure that refers to a corresponding generic block in the generic layout structure. This reference specifies that the common content portion associated with the generic block, is to be imaged in the specific block referring to that generic block.

3.4.2 Document interchange

The interchange representation of a document consists of a sequence of protocol elements:

- document profile descriptor
- layout descriptor
- text units

Apart from these three types for the document there exists a presentation capabilities descriptor which is used to provide the negotiation mechanism. Descriptors and text units are composite data elements consisting of a sequence of subordinate data elements and elementary data items. Within the subordinate data elements, further sequences of subordinate data elements and/or elementary data items can be nested many times. The elementary data items contain basic data types such as numbers, character strings, or bitstrings.

The *presentation capabilities descriptor* consists of four main parts:

- a data element which represents the basic terminal characteristics (see section 3.2.1),
- a data element which represents the interchange format,
- a sequence of data elements which represents the non-basic terminal capabilities,
- a sequence of data elements which represents the non-basic structural capabilities (e.g. number of objects per page). This part is not used in FAX 4 application.

The *document profile descriptor* also consists of four main parts:

- a data element which represents the reference to the generic layout structure of the document,
- a data element which represents the reference to the specific layout structure of the document,
- a sequence of data elements which represents the presentation capabilities a terminal must provide to be able to handle the document,
- an information field, which is for further study.

A *layout descriptor* represents a specific or generic layout object and its attributes. Attributes are parameters that specify characteristics of and relationship between

the objects and the content portions. The attributes belonging to the layout objects will be discussed in section 3.4.4.

The *text units* represent portions of document content and the associated attributes. A text unit consists of two main parts:

- a sequence of data elements representing the attributes of the portion of document content (see section 3.4.4),
- an information field consisting of one or more data elements that represent the portion of document content concerned. It always conveys a homogenous type of graphic elements (a mixture is not allowed).

Between descriptors and between descriptors and text units there exist *relationships*. These relationships are normally represented by structured identifiers or references. There are three types of relationships (see also Figure 3.9):

1. hierarchical relationship between objects. The relationship is represented by a structured identifier.
2. correspondence between a specific object and a generic object. In this case the relationship is represented by a reference.
3. correspondence between objects and contents. The relationship is represented by a structured identifier within the text unit. References in the descriptor of the page or block concerned to that text unit may be used in addition.

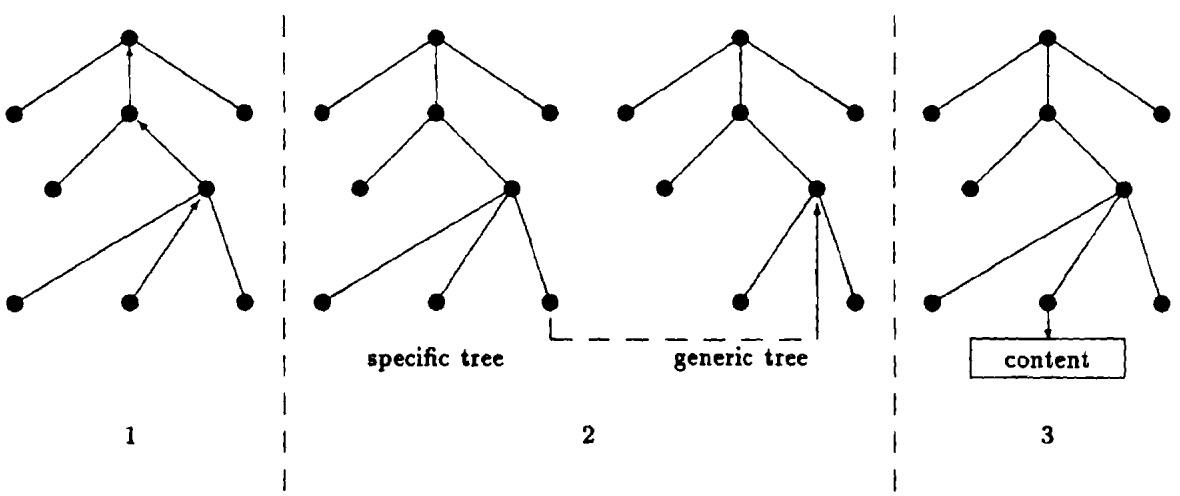


Figure 3.9: *Relationships between descriptors and between descriptors and text units.*

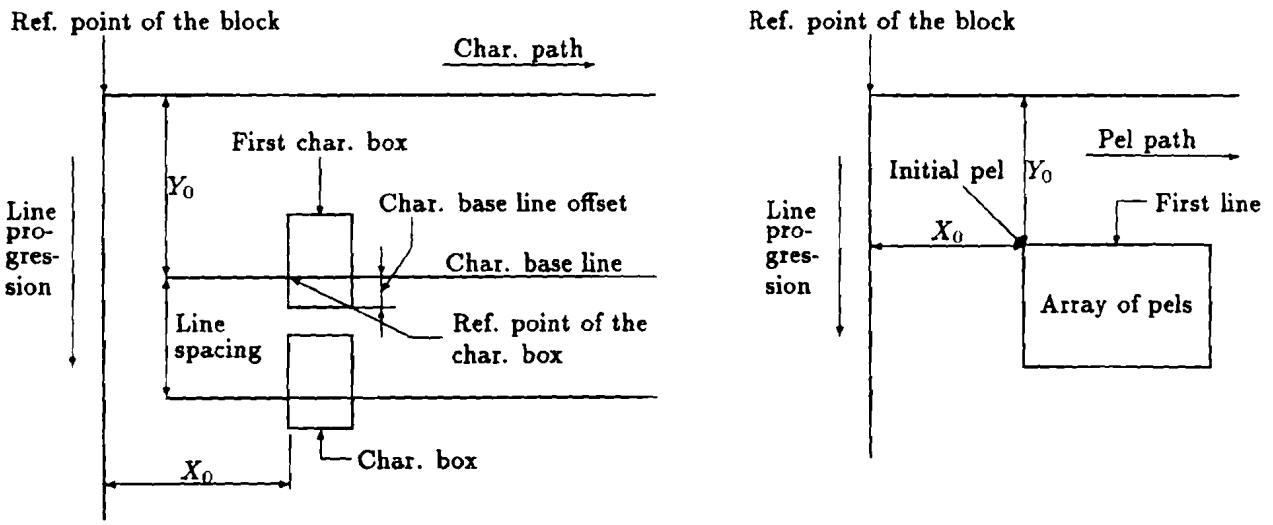
3.4.3 Text positioning

For positioning of layout objects, a Cartesian coordinate system, which has its origin in the top left corner of the page, is used. This means that objects are positioned in the fourth quadrant. The positions will be given non-negative values, in BMU[4], along the X-axis and Y-axis. The reference point for positioning layout objects is the top left corner of that object.

A layout object immediately below the level of page is positioned in absolute page coordinates, while all objects subordinate to that level use relative positioning.

To position text within blocks, distinction must be made between character coded and facsimile coded text (only the latter is used by FAX 4).

In case of character coded text it is assumed that each character is contained in a character box. In Figure 3.10 all the important dimensions and their relations for positioning of character coded and facsimile coded text are given.



X_0: Horizontal initial offset, Y_0: Vertical initial offset

Figure 3.10: *Positioning of character boxes and pels.*

The position of the first line is established from an initial point, specified by the coordinate distances from the reference point of the layout object to the reference point of the initial graphic element on that line. This, in conjunction with the graphic element path and the line length, specifies the positioning of the first line. Each subsequent line is positioned relative to the preceding line. The line progression direction is orthogonal to the graphic element path direction.

[4]BMU = basic measurement unit, i.e. $1/1200 \times 25.4mm$

3.4.4 Attributes

This subsection describes the attributes used by FAX 4.

The attributes of layout objects

The attributes of the layout objects can be classified in four categories:

- identification attributes,
- structure attributes,
- position attributes and
- presentation attributes.

Two identification attributes can be distinguished:

Object type: a mandatory attribute specifying whether the the object concerned is a document, page set, page, frame or block (in FAX 4 only document and page is a possible value for this attribute).

Object identifier: a non-mandatory attribute used for identifying an object uniquely and for referring to that object. The identifier consist of a sequence of numbers of which the first one begins with a 0 in case of a generic layout object, and a 1 in case of a specific layout object.

The structure attributes are listed below:

Reference to subordinate objects: used by all objects except those at the lowest level of the hierarchy. The attribute is a non-mandatory attribute for the document (root) and not applicable to the page.The contents is a sequence of decimal-coded integers specifying the subordinate objects at the next lower level of the hierarchy.

Reference to content portion: used by objects at the lowest level of the hierarchy. The attribute is a non-mandatory attribute for the page and not applicable to the root. The attribute specifies which content portions are associated to the corresponding page. In case more than one content portion is associated to a page, an ordering among these content portions is specified. The value of this attribute is a sequence of one or more integers, each corresponding to a content portion associated to the page. This integer, corresponding to a content portion, is the last integer of the attribute "content portion identifier".

Default value list: specifies a set of attribute value lists that are applicable as defaults to subordinate objects of designated object types. Each list must identify the object type to which it is to be applied and has as contents a set of attribute data elements that specify the default values to be applied. It is a non-mandatory attribute for the document (root) and specifies default values for the page in case of FAX 4 application. These defaultable attributes can be: presentation attributes and dimensions.

The position attributes consist only of:

Dimensions: only used for pages, frames and blocks. It is a defaultable attribute for the page. It consists of a pair of dimensions that specify the size of the object.

The presentation attributes are applied to the lowest level of the hierarchy (i.e page or block). The attributes depend on the *content type*. This content type identifies the type(s) of graphic elements forming the content of the object. As known there are two types, the character box elements and the facsimile graphic elements. Only the facsimile graphic element attributes are discussed, since only these are important for FAX 4 application.

Image orientation attributes. These attributes are:

- pel path specifies the direction of the progression of successive pels along a line, relative to the horizontal axis of the page in case of FAX 4 application,
- line progression specifies the direction of the progression of successive lines, relative to the pel path.

Image dimension attributes. There is only one attribute:

- pel transmission density: specifies the resolution, i.e. number of pels/unit of length.

The attributes of content portions

The attributes of the content portions are the content portion identifier, the type of coding, coding attributes, and alternative graphic rendition. These attributes are discussed in this subsection.

- The *content portion identifier* is a non-mandatory attribute that identifies a content portion uniquely within the context of the document. The identifier is used to refer to that content portion. The value of the attribute is a sequence of integers, of which the first integers indicate the page to which the content

portion is associated. The last integer identifies the content portion uniquely among the set of content portions, associated to that page. The first number has the same function as the first number of the object identifier.

- The *type of coding* is a defaultable attribute specifying the coding used to represent the content. For FAX 4 application the value of this attribute is an ASN.1[5] object identifier indicating a coding method as specified in CCITT Recommendation T.6.

- The *coding attributes* provide additional parametric information used in encoding/decoding the content portion. In case of character box elements it is for instance used to indicate an extended character repertoire. In case of photographic elements the following attributes are defined:

  - number of pels per line is a defaultable attribute that specifies the number of pels on each line within a content portion. Table 3.1 gives the number of pels per line for the relevant resolutions.

  - number of discarded pels is a defaultable attribute that specifies the number of discarded pels at the beginning and end of each line.

  - compression is a defaultable attribute indicating the presence of the code extension technique for the uncompressed mode. The value of this attribute can be 'compressed' or 'uncompressed'.

The default attributes

Now that the attributes of the layout objects and the content portions have been discussed, there still remain some attributes that need more attention. These are the default attributes. When an attribute, for example the positioning attribute of the layout objects, is specified at higher levels of the hierarchy, then this attribute is interpreted as default value for the lower levels. To determine the attributes of a specific layout object, the priority order is:

1. attributes specified explicitly in the specific layout object concerned,

2. attributes specified explicitly in the corresponding generic layout object,

3. default attributes specified in the specific object at the next higher level or its corresponding generic layout object,

4. the default values of the CCITT Recommendations.

[5] ASN.1 = Abstract Syntax Notation 1

3.4.5 Communication concepts

This section describes the use of the session layer by the DIP and the exchange (and invocation) of presentation capabilities.

The use of the session service is summarized in Table 3.6. This table shows which protocol elements are mapped on which session service primitives. These session service primitives are mapped on the commands and response described in section 3.3.1. For instance, the protocol element presentation capabilities descriptor is transferred to the session layer, during negotiation, using the parameter "user data" of the S-primitive S-CONNECT. This information is mapped on the Session User Data (SUD) parameter of the session command CSS.

Purpose	SS primitives	Protocol elements	Transfer of protocol element
Assignment of a presentation capabilities negotiation to the session connection and session capabilities exchange	S-CONNECT and/or S-CAPAB DATA	Presentation capabilities descriptor	S-CONNECT and/or S-CAPAB DATA
Assignment of a presentation capabilities indication / invocation to the session activity begin	S-ACT BEG	Presentation capabilities descriptor	S-ACT BEG
Use of session normal data	S-DATA	Doc. profile descr. Gen layout descr. Spec. layout descr. Text unit	S-DATA

Table 3.6: *Use of the session service by the document interchange protocol.*

In section 3.3 it was already pointed out that some of the session/document commands play an important role in the negotiation or invocation of the presentation capabilities. Here we will take a closer look at this negotiation process and the invocation process. The presentation capabilities attributes mentioned, are conveyed in the session user data (SUD), using the presentation capabilities descriptor protocol element.

The presentation capabilities are *negotiated* as follows:

- The CSS/RSSP command shall only indicate in its SUD which interchange format(s) and basic terminal capabilities are available as receiving capabilities of the sender of the command/response.
- The CDCL command indicates in its SUD a list of the optional receiving capabilities that may be needed by the sender of the command at the receiver.
- The RDCLP indicates in its SUD the optional presentation capabilities available.

The *invocation* is done by the CDS or CDC command. The presentation capabilities which are required for the document should be included in the SUD. The sender only sends documents of which the receiver has indicated it is capable of handling.

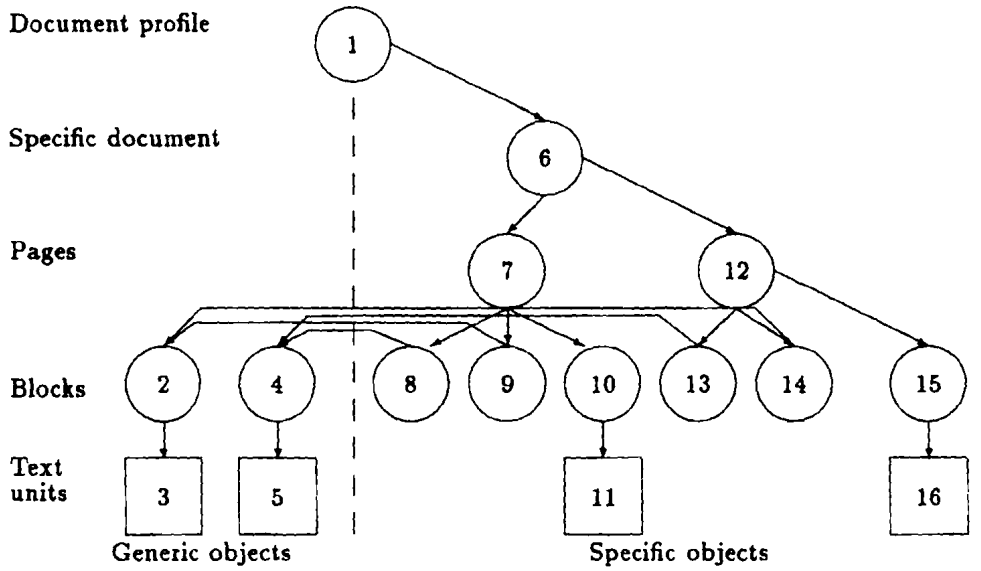


Figure 3.11: *Transmission sequence of protocol elements.*

The attributes used in the negotiation/invocation are the same as the attributes of the presentation capabilities descriptor (see section 3.4.4).

The CSUI/CDUI commands are used for the transfer of the document profile descriptor, the generic layout descriptor, the specific layout descriptor and the text units.

3.4.6 The application rules

The transmission of the description of a layout structure follows the natural order, i.e. the description of the tree structure. The transmission of the protocol elements is given in Figure 3.11.

Two Text Image Formats can be distinguished:

TIF0, which is used for documents that consist of photographic elements (facsimile) and

TIF1, which is used for documents that consist of photographic and character box elements (mixed-mode).

FAX 4 uses TIF0, which is defined with a specific layout structure consisting of:

- document descriptor

- page descriptor(s)
- text units.

This means that there is no document profile used by FAX 4 application. In appendix A the ASN.1 descriptions of the protocol elements used by TIF0 are given.

References

[5], [6], [7], [8], [9], [10].

Chapter 4

Facsimile group 4, the 1988 CCITT Recommendations

4.1 Introduction

This chapter discusses FAX 4 as defined in the 1988 CCITT T-Series Recommendations[1] and in report R2 of CCITT study group VIII, [12]. Like the 1984 recommendations, the 1988 recommendations classify the document facsimile apparatus in the same four basic categories, see section 3.1.

There is however a difference in the approach of the telematic services on the application layer of the OSI model, between the 1984 and 1988 recommendations. Where in the 1984 recommendations all services were specified separately, in the 1988 recommendations there is chosen for an integrated approach of all the telematic services such as FAX 4, mixed mode of service and videotex. This integrated approach of the telematic services is called DATAM[2]. DATAM consists of standards specifying the architecture of documents called ODA[3], a standard for the interchange of documents called ODIF[4] and standards specifying the document transfer services and protocols called DTAM[5]. In Figure 4.1 the structure of DATAM is clarified.

In Appendix B a description of DATAM is given. In section B.1 a study is made of DTAM (Recommendation series T.430), and in section B.2 the general concepts of ODA (Recommendation series T.410) are discussed. In this chapter only that

[1]The relevant recommendations are: T.411, T.412, T.414, T.415, T.417, T.431, T.432, T.433, T.503, T.521 and T.563

[2]DATAM = Document Architecture Transfer and Manipulation

[3]ODA = Open Document Architecture

[4]ODIF = Open Document Interchange Format

[5]DTAM = Document Transfer And Manipulation

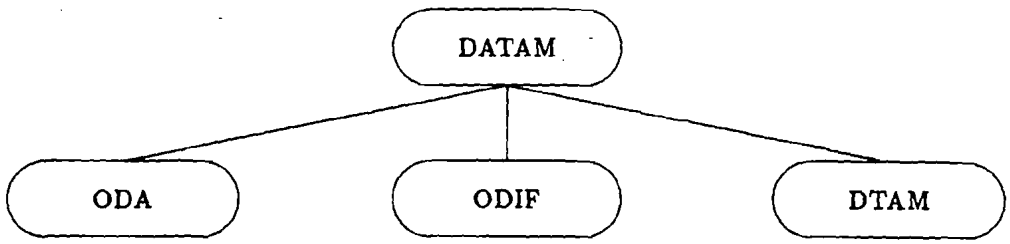


Figure 4.1: *DATAM structure*

part of DATAM that is of interest for FAX 4 is discussed. In section 4.2 the general aspects of FAX 4 (T.563) are given. In section 4.3 both the communication (T.521) and document (T.503) application profile, which are used to select the relevant parts of DTAM and ODA respectively, for FAX 4 application are described. And finally in section 4.4 the various attributes and the interchange format are discussed.

In Appendix F contradictions between ODA and FAX 4 and contradictions between DTAM and FAX 4 are summarized. These are problems within the 1988 T-series of recommendations, encountered during the study of these recommendations.

4.2 Terminal characteristics of FAX 4

In this section the terminal characteristics of FAX 4 as specified in the CCITT Recommendation T.563 are discussed. These characteristics differ only at some points from the 1984 terminal characteristics, see section 3.2. Only the differences are discussed.

The same three classes as mentioned in section 3.2.1 can be distinguished. All these classes must provide the basic characteristics/functions mentioned in the same subsection. There are however same differences, which are discussed below.

- FAX 4 terminals shall be capable of handling:
  1. the communication application profile specified in T.521 and discussed in section 4.3.1;
  2. the document application profile specified in T.503 and discussed in section 4.3.2.
- The requirements for the pel transmission density are given in the Table 3.1. The 1988 recommendations also specify pel transmission densities for Japanese Legal and Japanese Letter. These are given in Table 4.1

	Japanese Legal	Japanese Letter
Resolution (pels/inch)		
200	2018/2866[1]	1728/2024
240	2458/3439	2074/2428
300	3072/4299	2592/3035
400	4096/5732	3456/4047
Scan line length (mm) (P)	260.10	219.46
Paper width (mm) (Q)	257	182
$P - Q$	3.10	37.46
Nominal paper length (mm)	364	257

[1]Number of picture elements along a scan line/Nominal number of scan lines per page

Table 4.1: *Pel transmission density for Japanese Legal/Letter.*

- The use of optional functions can be negotiated during a handshaking procedure in the communication application profile,

4.3 Application profile for FAX 4

The communication application profile and document application profile for FAX 4 are discussed in this section. These profiles are specified in the CCITT Recommendations T.521 and T.503 respectively.

4.3.1 Communication application profile of FAX 4

The general aspects of the communication application profile are described in Appendix B.1.2. In this subsection the communication application profile of FAX 4 is described.

The communication application profile must specify:

- a service class;
- the functional units and
- communication support functions.

FAX 4 uses the communication application profile BT_0. This communication application profile uses DB_0 as the combination of the service class and the communication support functions. From Table B.1 it is seen that this means:

a: the service class = direct document bulk transfer;

b: communication support functions = direct mapping to the session service.

The functional units used by BT_0 are:

1. Association use control;
2. Capability;
3. Document bulk transfer;
4. Token control;
5. Exception report and
6. Reliable transfer mode 1[6].

The DTAM services of these functional units are listed in Table 4.2 below.

functional unit	service
association use control	D-INITIATE D-TERMINATE D-U-ABORT
capability	D-CAPABILITY
document bulk transfer	D-TRANSFER
token control	D-TOKEN-PLEASE D-CONTROL-GIVE
exception report	D-U-EXCEPTION-REPORT D-P-EXCEPTION-REPORT
reliable transfer	D-TRANSFER session service primitives

U = User, P = Provider

Table 4.2: *DTAM services for each functional unit.*

The direct mapping to the session service as the communication support functions was mentioned. But what does this mean?

Before the mapping to the session service is explained, distinction between DTAM and session services has to be made and the concept of protocol data units (PDUs), see [28], has to be described.

The DTAM services were mentioned in Table 4.2, and start with "D-". The session services used are those specified in Recommendations X.215 and T.62bis. However,

[6]There are two reliable transfer modes. Mode 1 where secure transfer is under the responsibility of DTAM, but the resumption of an interrupted transfer is under the responsibility of the DTAM user. Mode 2 where secure transfer is completely under the responsibility of DTAM (including resumption of an interrupted transfer)

at the moment it is not yet possible to use this combination of recommendations. This is caused by the incomplete specification of CCITT recommendation T.70bis[7]. This recommendation, in combination with X.214 has to provide for the transport layer. To solve this problem, recommendation T.62 is used to provide for the session layer. A problem caused by this solution is the use of commands and responses by T.62 instead of PDUs. However, these commands and responses can be seen as "PDUs".

PDUs are used to communicate between peer entities (i.e. two layers at the same level in the OSI reference model). These PDUs can convey service primitive parameters in their parameters. The PDUs of the application layer are called APDUs. Those of the session layer are called SPDUs.

Now the direct mapping to the session layer can be discussed.
First of all direct mapping to the session layer means that FAX 4 works in transparent mode (so, no presentation layer).
Secondly, it means that D-services are mapped onto the APDU parameters. These again are mapped onto the session-services, usually as user data, but not always. The session-services are mapped on the SPDU parameters.
In Figure 4.2 an example of the mapping is given.

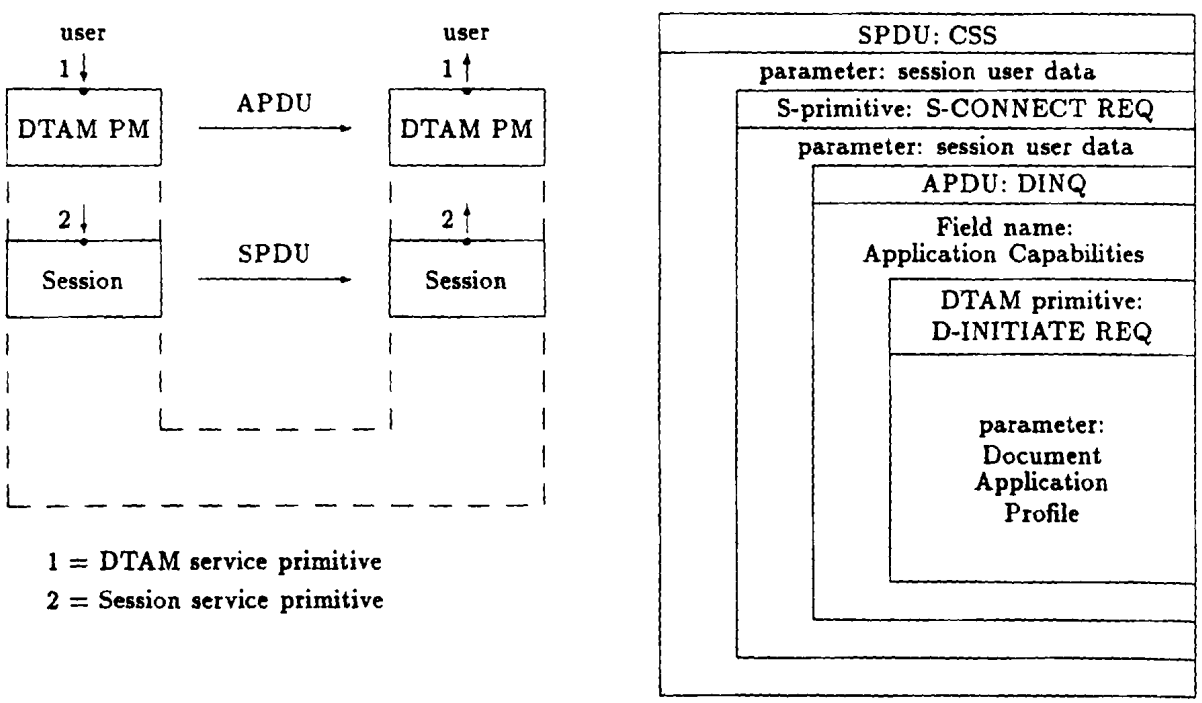


Figure 4.2: *Example of the mapping procedure of FAX 4*

The left part of Figure 4.2 shows the relevant part of the OSI model. Since transparent mode is used, there will be no presentation layer. The right part of Figure 4.2

[7]During the current study-period (1989-1992) of CCITT this recommendation is being specified.

must be read as follows: The parameter Document Application Profile of the DTAM service primitive D-INITIATE-REQ is mapped onto the Application Capabilities field of the APDU DINQ, etc.

4.3.2 Document application profile of FAX 4

The general aspects of the document application profile are given in Appendix B.2.7

The document application profile must specify:

- document architecture level;
- content architecture level;
- document profile level and
- interchange format class.

In case of FAX 4 these features have the following values (in the same order):

- Formatted Document Architecture (FDA);
- formatted raster graphic content architecture level;
- document profile is mandatory (discussed in the next section);
- interchange format class is ODIF class B (discussed in the next section).

What does this mean?
In section B.2.5 it was seen that the document architecture class FDA includes a document profile, a specific layout structure and optionally it can include presentation styles and a generic layout structure. This means that FAX 4 does not use logical structures (no logical components). Furthermore, the generic layout structure and presentation styles are not applicable and the specific layout structure has only two hierarchical levels:

1. document layout root;
2. page.

The absence of presentation styles is not explicitly (like the logical and generic layout structures) specified in recommendation T.503. However if this constituent is used, it has to be specified in the document profile. The document profile of FAX 4 is specified in table 1/T.503 which does not mention the presentation style constituent.

4.4 The document interchange

In this section it is shown how the document is transmitted. First, we take a look at the content architecture and document profile used by FAX 4. Then the various ODA attributes used by FAX 4 are discussed. Finally, ODIF class B, as used by FAX 4, is described.

4.4.1 Formatted raster-graphic content architecture

The content architecture used by FAX 4 is the formatted raster-graphic content architecture.
The content of a basic component that conforms to a raster-graphic content architecture represents a two dimensional pictorial image in the form of a rectangular two dimensional array of pels. The formatted raster-graphic content architecture may only be used in formatted form documents.

For positioning pels, a coordinate system with its horizontal and vertical axis parallel to the horizontal and vertical axis of the page coordinate system is used. The origin of the page coordinate system is the top left corner of the page. The horizontal axis coincides with the top edge of the page, the vertical axis coincides with the left side of the page. To identify points or to specify lengths within a basic layout object, Scaled Measurement Units (SMU)[8] are used. In Figure 4.3 the important dimensions and their relations for positioning pels are given.

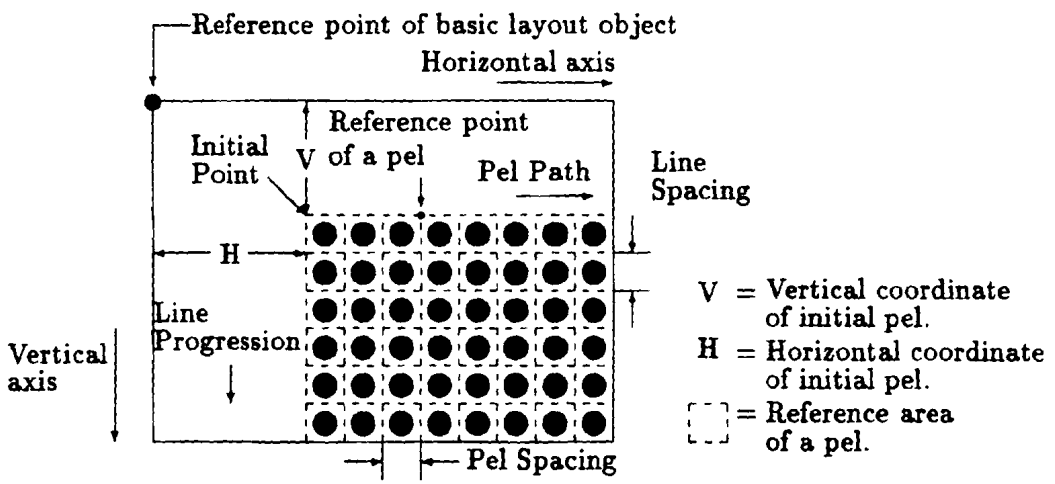


Figure 4.3: *Position of pels in basic object.*

[8] 1 SMU = m/n BMU, 1 BMU = $1/1200 \times 25.4mm$. m/n = the unit scaling factor, which is in case of FAX 4 equal to 1

The attributes that apply to the formatted raster-graphic content architecture can be divided into presentation attributes and content portion attributes.

- The presentation attributes are:
  - Pel path;
  - Line progression;
  - Pel transmission density.
- The content portion attributes are:
  - Number of pels per line;
  - Number of discarded pels;
  - Type of coding and
  - Compression.

These attributes were discussed in section 3.4.4.

The content imaging process of the formatted raster-graphic content architecture proceeds as follows:
The pels of the interchanged pel array which remain after the pels at the beginning of each line, specified by the attribute "Number of discarded pels", have been subtracted, are imaged. Any parts of a raster-graphic content portion which extends beyond the boundaries of the basic layout object are not imaged.

4.4.2 Document profile for FAX 4

The purpose of the document profile is to provide information, by means of attributes, for the document as a whole. It may be interchanged without the document body.

In case of FAX 4, the attributes of the document profile are:

- Document profile descriptor (set of attributes)
- Specific layout structure
- Document characteristics (set of attributes)
- Document application profile
- Document architecture class
- Non-basic document characteristics (set of attributes)
- Page dimensions

- Raster-graphic coding attributes (set of attributes)
- Compression
- Raster-graphic presentation features (set of attributes)
- Pel transmission density

The relationships between the attributes are clarified in Figure 4.4. These attributes are discussed in the next section.

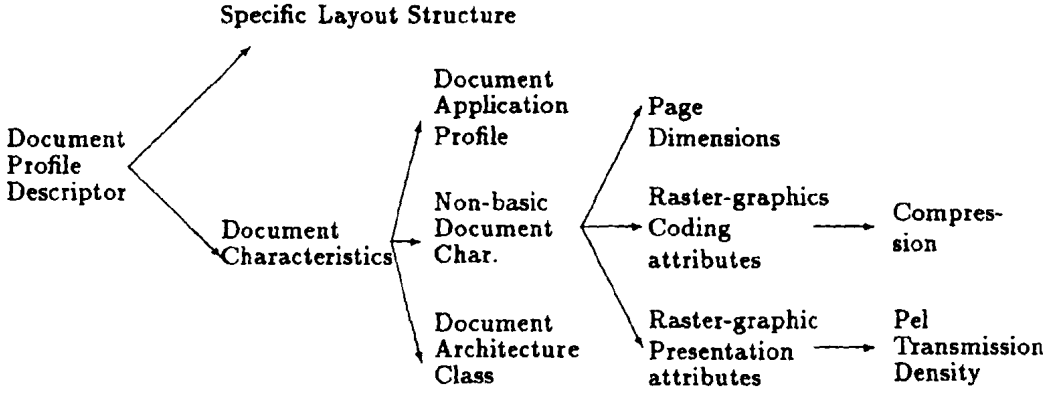


Figure 4.4: *Relationships between the various document profile attributes.*

4.4.3 ODA attributes applicable to FAX 4

In this section the various ODA attributes used by FAX 4 application are discussed. In appendix A these attributes are summarized together with their (possible) values.

Document profile attributes

Here the attributes mentioned in the previous section and belonging to the category "document profile attributes" are discussed.

Document profile descriptor is a mandatory set of attributes and it contains the following attributes for FAX 4:

Specific layout structure is a mandatory attribute and is used if and only if the document contains any layout object descriptions. The value of this attribute is 'present'.

Document characteristics is a set of attributes, which is mandatory and contains the following attributes for FAX 4:

Document application profile is a mandatory attribute that specifies the document application profile that pertains to the document. In case of FAX 4 class 1, the value of this attribute is the integer 2.

Document architecture class is a mandatory attribute that indicates the document architecture class of the document. In case of FAX 4 its value is 'formatted' (FDA).

Non-basic document characteristics is a set of non-mandatory attributes that specify possible non-basic values for these attributes. In case of FAX 4 this set contains the attributes:

Page dimensions is a non-mandatory attribute that specifies the non-basic values of the attribute "dimensions" of layout objects of type page. The value consists of one or more pairs of page dimensions (see also the layout object description attribute "Dimensions")

Raster graphics coding attributes is a non-mandatory set of attributes, which contains the following attribute:

Compression is a non-mandatory attribute that specifies if the code extension technique for uncompressed mode is present in a content portion. The value (if specified) is uncompressed (see also the content portion attribute "Compression").

Raster graphic presentation attributes is a non-mandatory set of attributes, which contains the following attribute:

Pel transmission density is a non-mandatory attribute specifying the pel transmission density, pel spacing and line spacing (see also the presentation attribute "Pel transmission density").

Component description attributes

Since FAX 4 documents have no logical structure, only shared and layout attributes are applicable. Furthermore, FAX 4 documents do not have a generic structure. So only shared attributes and layout attributes applicable to layout object descriptions are discussed. The only layout objects are the layout document root (in the following discussion abbreviated to root) and the page (basic), as mentioned in section 4.3.2.

Most of the attributes mentioned below were already described in section 3.4.4. If there is no explanation of an attribute, reference is made to that section.

Shared attributes:

There are four shared attributes applicable to layout object descriptors. These attributes are:

- Object type;
- Object identifier;
- Content portion (in section 3.4.4 this attribute is called "Reference to content portion");
- Default value.

Layout attributes:

There are two layout attributes applicable to layout object descriptors. These attributes are:

- Presentation attributes are defaultable for the page and not applicable to the root. They depend on the content architecture, which is the raster-graphic content architecture. These attributes are discussed at the end of this section.
- Dimensions is a defaultable attribute for the page and not applicable to the root. For FAX 4 the attribute consists of two parameters: "horizontal dimension" and "vertical dimension". The parameter "horizontal dimension" is represented by one subparameter: "fixed dimension". The parameter "vertical dimension" includes two subparameters: "fixed dimension" or "variable page height". The values of these parameters are integers specifying the width and height in BMU.

Presentation attributes:

The presentation style attributes applicable to FAX 4 are the set of presentation attributes mentioned below:

- Content type;
- Pel path;
- Line progression;
- Pel transmission density (see section 3.4.4). Since there is a correspondence between the pel spacing, line spacing and resolution, shown in Table 4.3, the attribute also specifies the pel and line spacing. Its basic value is 200 ppi. However, in the document profile this value can be changed, using the Non-basic document characteristics attributes[9]

Layout style attributes and *Presentation style attributes* are not used by FAX 4 application.

[9]i.e. the attribute "pel transmission density"

pel spacing and line spacing in BMU	resolution in number of pels per inch
6	200
5	240
4	300
3	400
2	600
1	1200

Table 4.3: *Correspondence between pel spacing, line spacing and resolution.*

Content portion attributes:

The content portion attributes applicable to FAX 4 application are:

- Content information is a mandatory attribute that is used for interchanging the actual content of the document. The value of this attribute is an octet string representing a pel array encoded according to the value of the attribute "Type of coding";
- Content portion identifier;
- Type of coding.

and the raster-graphic coding attributes:

- Number of pels per line;
- Compression;
- Number of discarded pels.

4.4.4 Open document interchange format

The constituents (mentioned in Appendix B.2.3) and the attributes (mentioned in section 4.4.3) of a document are represented by a hierarchy of data structures and data items. These structures and items are specified using ASN.1. The data structures are called *interchange data elements*. All the descriptors, except the content portion descriptors, consist of simple and composite data items. These represent the attributes of the descriptor concerned. A content portion is represented by one text unit. This text unit consists of two fields:

a) **an attribute field,** i.e. a data structure consisting of simple and composite data items representing the attributes of the content portion concerned.

b) **an information field,** i.e. a data structure consisting of a data item or a set of data items representing content elements making up the content portion concerned.

These elements are ordered in accordance with certain rules, which are specified in the Open Document Interchange Format (ODIF). Within ODIF two interchange format classes can be distinguished, class A and class B. In case of FAX 4 application **class B** is used.

This class consists, in case of FAX 4, of the following interchange data elements (interchanged in the order given):

- document profile descriptor
- layout object descriptor (root and pages) and associated text units.

Within the last group the order of the objects is equal to the sequential order as specified in section B.2. However, each descriptor of a basic layout object (page) is immediately followed by the associated text units. This means for FAX 4 that every page in a data stream is followed by its associated content portions.

In all cases, a data stream contains one and only one document profile descriptor. This document profile descriptor is always the first interchange data element in the data stream.

In appendix A the ASN.1 description of the data structures is given.

References

[11], [13], [14], [15], [16], [12].

Chapter 5

Compilation of incompatibilities

5.1 Introduction

In this chapter, problems, which may arise during interworking between FAX 4 implementations according to the 1984 CCITT Recommendations, see [5], and FAX 4 implementations according to the 1988 CCITT Recommendations, see [11] and [12], are discussed. The study will be focussed on the upper two layers of the OSI Reference Model. Since FAX 4 does not use the presentation layer, these two layers are the application layer and the session layer, see Figure 5.1

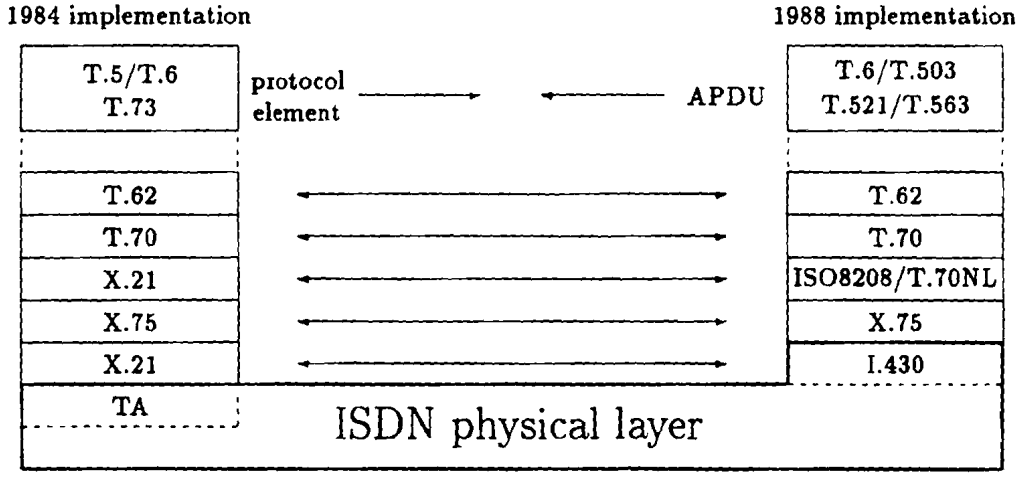


Figure 5.1: *The B-Channel communication system* .

A FAX 4 implementation according to the 1984 (1988) CCITT Recommendations is abbreviated to a 1984 (1988) implementation. The communication between the two implementations will be over an ISDN. The communication system for the B-Channel is given in Figure 5.1. Since the 1984 implementation has no interface to

the ISDN, a terminal adapter (TA) will be necessary.

Figure 5.1 shows that the session service and protocol is specified in recommendation T.62. This recommendation however does not specify the service primitives in detail, i.e. only the primitives used are mentioned, but no parameters of these primitives are specified. To solve this problem, recommendation T.62bis, which specifies the session services in detail, is used. The only reason for this choice is the fact that although T.62bis is not a 1984 standard, it did not appear out of the blue.

5.2 Communication aspects

One of the first things noticed by looking at Figure 5.1, is the use of APDUs by the 1988 implementation and the use of protocol elements by the 1984 implementation. The 1988 implementation also uses DTAM service primitives, which have no equivalent in the 1984 implementation. These differences may cause problems in the communication between the two implementations. In the next sections these differences are investigated. In Appendix C the establishment procedure between the two implementations is described.

5.2.1 APDUs versus protocol elements

Before the APDUs and protocol elements (see section 3.4.2) are compared, first an explanation of an APDU is given. An APDU is a minimum data element, containing one or more fields, interchanged between application entities. They are formed by the DTAM protocol machine and the DTAM service primitives. The DTAM service primitives used, were mentioned in section 4.3.1. Only a few of these service primitives form an APDU. The other DTAM service primitives are directly "mapped" onto the session service primitives. This is discussed in section 5.2.2. The APDUs formed are mentioned in Table 5.1 together with the initiating DTAM service primitives.

DTAM service primitive	APDU
D-INITIATE	DINQ and DINR
D-TERMINATE	DTEQ and DTER
D-ABORT	DAB
D-CAPABILITY	DCPQ and DCPR
D-TOKEN-PLEASE	DTP

Table 5.1: *DTAM service primitives and the APDU s they form.*

The APDUs DTEQ and DTER are implicitly used, i.e. the APDU may be send in a parameter of a session service primitive, but the receiver of this session service

primitive does not look at these parameters. Receipt of the session service primitive by the receiver provides enough information. In case of FAX 4 application these APDUs are not even send, because S-RELEASE does not use the parameter "User Data" to convey these APDUs, see [17]. Therefore, these APDUs do not cause any problems.

The APDU DAB is also used implicitly, and according to T.433 not send in case of transparent mode. Moreover, it can not be transferred, because S-U-ABORT has no "SS-User Data" in case of FAX 4 application. So, no problems are caused by this APDU.

The APDU DTP is used implicitly. It is not transferred in case of FAX 4 application, because the S-TOKEN-PLEASE primitive has, in that case, no "User Data" parameter to convey the APDU.

This means that only the ADPUs DINQ, DINR, DCPQ and DCPR may cause problems. These APDUs consist of several fields. The DINQ and DINR APDUs consist of the field:

- Application Capabilities consisting of the sub-parameters:
  - Document Application Profile and
  - Document Architecture Class.

The DCPQ and DCPR APDUs consist of the field:

- Application Capabilities consisting of the sub-parameters:
  - Document Application Profile,
  - Document Architecture Class and
  - Non Basic Document Characteristics.

How does the transmission of these APDUs proceed, when interconnecting a 1984 and a 1988 implementation?
During the association establishment procedure (detailed description is given in Appendix C), the contents of the D-INITIATE service primitive parameter "Application Capabilities" is mapped onto the DINQ APDU field "Application Capabilities". This APDU is mapped onto the user data of the S-CONNECT service primitive, which is mapped again onto the user data of the Command Session Start (CSS) "PDU". At the receiver side, this procedure proceeds in the opposite direction. The Document Interchange Protocol (DIP) of the 1984 implementation obtains the user data of S-CONNECT. This user data contains:

- Document Application Profile (used in the T.400 series) which is equal to the Basic Terminal Characteristics (used in T.73),
- Document Architecture Class (used in the T.400 series) which is equal to the Interchange Format (used in T.73),

According to section 3.4.5 this is exactly what is expected by the 1984 implementation.

In case of the D-CAPABILITY service primitive the communication proceeds likewise. Here no problems will arise because the "Non Basic Document Characteristics" of the T.400 series is equal to the "Non Basic Terminal Capabilities" of T.73, see section 3.4.5. The other sub-parameters were mentioned above.

Conclusions: The use of APDUs by the 1988 implementation causes no problems to the 1984 implementation. And vice verse do the protocol elements not cause any incompatibilities.

5.2.2 DTAM service primitives and T.73

Since the application layer is the top layer of the OSI Reference Model (it does not interface with a higher layer), there are no application service access points (ASAPs) and no application service primitives. This means that the DTAM service primitives can not be seen as "normal"[1] service primitives, which is clarified by Figure 5.2.

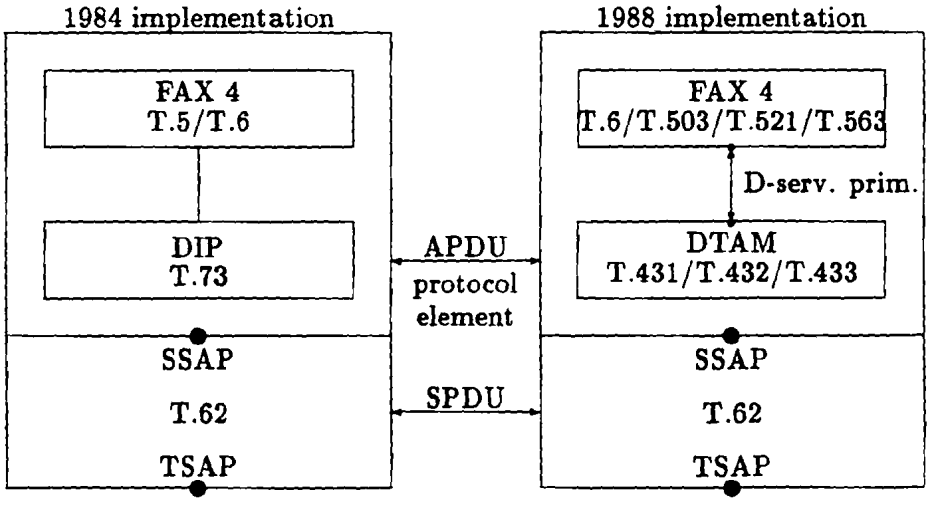


Figure 5.2: *Upper two layers of a facsimile group 4 implementation.*

[1]Like the OSI Reference Model describes.

This figure shows that the top layer of a 1988 implementation is subdivided into a specific-application-service-element (DTAM) and a user element (FAX 4). These elements communicate by means of the DTAM service primitives (so communication within a layer and not between layers). The 1984 implementation also shows a division of the top layer into the DIP and the FAX 4 application. Here no service primitives are used for the communication.

In case of FAX 4 application the mapping of the DTAM service primitives onto the APDUs is a one to one mapping, i.e. a parameter X of a DTAM service primitive is mapped onto a parameter Y of the APDU. However, not all the DTAM service primitive parameters are mapped onto an APDU. The ones which are not mapped onto APDUs are mapped directly onto the session service primitive parameters, determine a session service primitive parameter or are not mapped at all. The result is, that they are only important for the session layer and not for the application layer of the peer entity (=1984 implementation) unless they are mapped onto "User Data". These parameters are:

- D-INITIATE response "Result" parameter, which determines the S-CONNECT response "Result" parameter.
- D-TRANSFER request "Document Information" parameter (i.e. the actual document contents) is mapped directly onto one or more S-DATA request "User Data" parameter (segmenting the actual document data requires more S-DATA requests).
- D-TRANSFER request "Document Information Type" parameter is not mapped at all. Depending on its value it initiates a S-ACT-START request or S-ACT-RESUME request[2] service primitive.
- D-TRANSFER request "Document Reference Information" parameter is not mapped at all. It initiates a S-ACT-START request service primitive.
- D-TRANSFER confirm "Result" parameter is not mapped at all. It is initiated by a S-ACT-END confirm service primitive.
- D-U-ABORT can optionally use, according to T.521, the parameter "User Information". According to T.432 this parameter is absent in transparent mode. Whether the parameter is used or not is not important, since the primitive forms an DAB APDU. This APDU is implicitly used to form a S-U-ABORT (see previous section). This primitive may cause problems because it is not clear whether it is a confirmed service of not. This is discussed in section 5.2.3 (see also Appendix F).

The other DTAM service primitives (i.e D-TERMINATE, D-TOKEN-PLEASE, D-CONTROL-GIVE and D-P-EXCEPTION-REPORT) use no parameters in FAX 4

[2]Since the value of this parameter is always 'transfer a document from its beginning', no S-ACT-RESUME is used.

application. D-P-EXCEPTION-REPORT is driven by a S-P-EXCEPTION-REPORT. This means that the primitive is only used in an upward direction in the OSI Reference Model. D-CONTROL-GIVE initiates directly a S-CONTROL-GIVE service primitive. The remaining DTAM primitives use APDUs implicitly.

Conclusion: Since all information, that does not belong to the actual document information, is more or less mapped directly onto the session service primitives, the DTAM service primitives cause no problem to the 1984 implementation. However, D-U-ABORT may cause problems, because of its ambiguous specification.

5.2.3 Use of the session layer

In this section a comparison is made of the use of the session service by both implementations. Two differences between the use of the session service by both implementations can be distinguished:

1. S-U-ABORT conf. is not used by the 1988 implementation;
2. The 1988 implementation has no session resume procedure.

Ad 1: According to [11, T.433 section 6.4.3.2.2], the responding DTAM-PM issues a D-U-ABORT ind. and a S-U-ABORT resp. after receiving a S-U-ABORT ind. According to the state tables in the same recommendation, only a S-U-ABORT req. and ind. is used, see Appendix F. In recommendation T.521 annex A, S-U-ABORT resp. is not used, but a S-U-ABORT ind. is initiated by the transport provider. This is strange and very likely to be wrong. In recommendation X.215 S-U-ABORT is an unconfirmed service. In recommendation T.62bis it is not stated whether the service is confirmed or not, but since this recommendation is based on X.215/X.225 an unconfirmed service is assumed.

The above shows that this service primitive causes some problems. Since a response primitive without a confirm primitive is not possible according to the OSI reference model and since a user abort initiated by the provider is hardly to be considered as being right, at this point it seems best to choose for an unconfirmed S-U-ABORT service. This choice is strengthened by the fact that recommendation T.432 specifies an unconfirmed D-U-ABORT service, which initiates the S-U-ABORT service. However, this will not only cause problems with the 1984 implementation, which uses a confirmed S-U-ABORT service, it will also cause problems with the session protocol, see [5, T.62].

If the 1984 implementation sends the abort, then it will wait for a S-U-ABORT conf, which is initiated by the receipt of the "PDU" Response Session Abort Positive (RSAP). If after 4 seconds this "PDU" has not been received, then the transport connection is terminated (i.e. a provider abort) by the 1984

implementation. In the other case, where the 1988 implementation sends the abort, the 1984 implementation will respond with a S-U-ABORT resp. This means that the DTAM-PM receives a S-U-ABORT conf service primitive, which is not defined for the PM.

Ad 2: The 1984 implementation is able to resume transfer after an error[3]. This is only possible when at least one checkpoint has been confirmed. The 1988 implementation is not able to resume transfer, because T.521 states that the D-TRANSFER parameter "Document information type" *always* has the value 'transfer of a document from its beginning'. So, the 1988 implementation does not use the session (T.62) resume service.

What does this mean for the interworking between the two implementations? Two situations can be distinguished:

1. The 1988 implementation is the sender;
2. The 1984 implementation is the sender.

In the first case there will be no resume procedure after a S-ACT-INTERRUPT. The 1984 implementation is however waiting for a S-ACT-BEG (CONTINUE)[4] ind, but receives a S-ACT-BEGIN (START) ind. This means that the transmission of the document starts again from the beginning. But what about the data already acknowledged? According to T.62 it is the responsibility of the receiver to discard any information which has been duplicated in the process of continuation of an interrupted service.

In case the 1984 implementation is the sender, more severe problems may arise. The 1984 implementation will start the resume procedure, while the 1988 implementation is waiting for a S-ACT-START ind. This results in an invalid intersection in the state tables of the DTAM-PM (see Figure 5.3).

According to T.433 annex B 3.1 now one of the following actions can occur:

- the DTAM-PM issues an appropriate internal event or
- the DTAM-PM issues both an D-P-ABORT ind. to the DTAM-user and a S-U-ABORT req. to its peer DTAM-PM.

In the latter case, the 1984 implementation receives a Command Session Abort "SPDU", and sends a S-U-ABORT ind. to its user. After receiving a S-U-ABORT resp. from its user, it sends a Response Session Abort Positive (RSAP) "SPDU" and goes into the idle state, waiting for S-CONNECT req.

Conclusions: The S-U-ABORT service primitive causes an incompatibility when used as an unconfirmed service by the 1988 implementation. Furthermore, the question remains whether this service can be used as an unconfirmed service since T.62 states that it is a confirmed service. The resume procedure is used by the

[3] As far as known, terminals available on the market do not have the capability to do so.

[4] In T.62 the term S-ACT-BEGIN (START) is used for the T.62bis/X.215 term S-ACT-START, and the term S-ACT-BEGIN (CONTINUE) is used for the term S-ACT-RESUME.

1984 implementation, but not used by the 1988 implementation. This may cause an incompatibility.

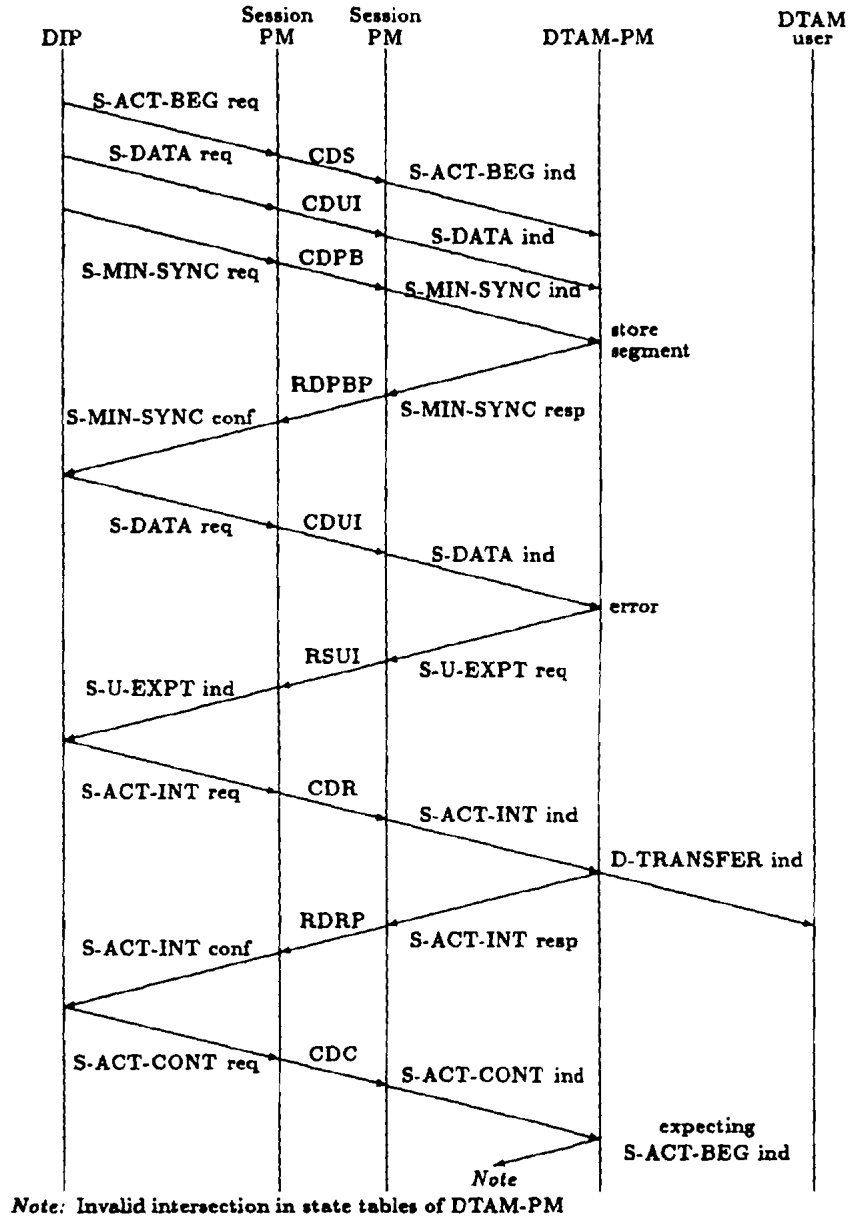


Figure 5.3: *Transfer resume procedure started by the 1984 implementation.*

5.3 Document aspects

In this section the compatibility of the document profiles and the document structures used by the different implementations is described. Furthermore, the various attributes used by both implementations are compared.

5.3.1 Document profiles

For the 1984 implementation no document profile descriptor is specified, see section 3.4.6. The 1988 implementation uses a document profile, see section 4.4.2, which consists of:

- specific layout structure and
- document characteristics consisting of:
  - document application profile;
  - document architecture class and
  - non-basic document characteristics.

At first glance this seems to cause an incompatibility between the two implementations. However, the document profile used by the 1988 implementation is not interchanged. Only the "document characteristics" of the document profile are interchanged. This is done by the D-TRANSFER parameter "Document Information". The DTAM-PM uses this information to generate the DTAM capabilities and convey them in the S-ACT-START parameter "User Data". At the receiving DTAM-PM, the document profile is reconstructed using these document characteristics.
This is also done in the 1984 implementation, where the presentation capabilities descriptor (= Document Characteristics) is interchanged at the beginning of the document transfer, using S-ACT-START.

ETSI states in [18] that the document profile attributes "content architecture class" and "interchange format class" are mandatory for the 1988 implementation. The same is stated in recommendation T.414 annex B (see Appendix F). These have no counterpart in the 1984 implementation and may therefore cause problems of incompatibility.

Conclusions: Since the document profile is not interchanged in the 1988 implementation, and the 1984 implementation does not use a document profile, there will be no incompatibility between the two implementations. Using the ETSI document however, see [18], makes the implementations incompatible.

5.3.2 Document structures

The 1984 implementation uses TIF0, which means that the structure consists of:

- document descriptor and

- page descriptor(s).

The 1988 implementation use FDA, which in this case consists only of:

- document layout root and
- page

Conclusion: Both implementations use the same document structure, so no incompatibility is cause by the document structures, although they use different names.

5.3.3 Attributes

In this section the attributes that show incompatibilities are discussed.

The attribute "Reference to subordinate object", which can be used by the document descriptor in the 1984 implementation, has no counterpart in the 1988 implementation. Normally this should be the attribute "subordinates". This attribute is however not specified as applicable to FAX 4 application.

When comparing the ASN.1 descriptions of the attribute "Dimensions" in TIF0 and ODIF Class B a difference is detected. In TIF0 both the subparameters of this attribute (i.e. 'horizontal' and 'vertical') can be of variable length. In ODIF Class B only the subparameter 'vertical' can be of variable length. The subparameter 'horizontal' must be of fixed length. In T.73 it is stated that the use of variable dimensions is application-dependent (e.g. the "unlimited page length" defined in T.5). In T.5 only the page length (not the page width) is mentioned as being variable (end of page is detected by the paper scanner). The page width is of fixed length. This means that no incompatibility is caused by this attribute.

The attribute "Number of discarded pels" can have as value any integer in the 1984 implementation. In the 1988 implementation its value is restricted to the values summarized in [11, T.503].

The attribute "Content information" in the 1988 implementation has as counterpart the information field "Text information". This causes no incompatibility, but one could ask the question whether this "Content information" is an attribute or not.

Conclusions: The attributes "Reference to subordinate object", and "Number of discarded pels" cause incompatibilities between the two implementations.

5.3.4 Interchange format

As mentioned in section 5.3.2, the 1984 implementation uses TIF0. The 1988 implementation uses ODIF class B. Comparing these interchange formats shows that they have the same structure. There are however two minor differences, not in the structure, but in the coding. Since both formats are described in ASN.1, they use the ASN.1 encoding rules. This implies that attributes have to be coded identically in both implementations to assure compatibility.

The two differences mentioned are:

- The attribute "pel transmission density" (sub-attribute of "non-basic-document-characteristics") in the document profile of the 1988 implementation is coded differently from the same attribute in the presentation capabilities (sub-attribute of "non-Basic-Terminal-Capabilities") of the 1984 implementation.
- The same holds for the attribute "compression" in the document profile of the 1988 implementation and in the presentation capabilities of the 1984 implementation.

The different coding is given in Table 5.2.

1988 coding	1984 coding
A2 non-basic-document-characteristics 0A length A4 ra-gr-presentation-features 03 length **82** pel-transmission-density 01 length 02 value (= 240 pels/inch possible) A3 ra-gr-coding-attributes 03 length **82** compression 01 length 00 value (=uncompressed)	A2 nonBasicTerminalCapabilities 0A length A4 presentationAttributes 03 length **8B** pelTransmissionDensity 01 length 02 value (= 240 pels/inch possible) A3 codingAttributes 03 length **80** compression 01 length 00 value (=uncompressed)
Total coding: A2 0A A4 03 82 01 02 A3 03 82 01 00	Total coding: A2 0A A4 03 8B 01 02 A3 03 80 01 00

Table 5.2: *Different coding of pel transmission density and compression.*

The receiver of this data will detect an error in the coding and shall abort the session. The 1988 implementation will send a S-U-Abort req to its peer entity and a D-P-Abort ind to its user. The 1984 implementation sends a S-U-Abort req.

Conclusions: The different ASN.1 coding of the attributes "pel transmission density" and "compression" causes an incompatibility between the two implementations.

5.4 Terminal aspects

Comparing the terminal aspects of FAX 4 implementations means comparing section 3.2.1 with section 4.2

5.4.1 Japanese page sizes

The 1988 implementation is capable of handling Japanese Legal and Japanese Letter. The 1984 implementation has no such capabilities. This means that whenever a 1984 implementation is the sender this difference will cause no problems. But what if the 1988 implementation is the sender?

During the negotiation procedure of optional functions at the beginning of an association, the page dimensions are determined.
In case of Japanese Legal the dimensions are 12141×17196 BMU. At the 1984 implementation this size can only be covered by ISO A3 (14030×19840 BMU) page size.
In case of Japanese Letter the dimensions are 8598×12141 BMU, which is covered by ISO A4 (9920×14030 BMU) page size.

Conclusion: Use of Japanese page sizes by the 1988 implementation causes no problems to the 1984 implementation, since these sizes are covered by other page sizes. If these non-standard page sizes are not available, then the association is ended.

5.4.2 Negotiation procedure

In the previous section the negotiation procedure of optional functions was mentioned. In the 1984 implementation this procedure takes place in the end-to-end control procedure, i.e. in the session layer (T.62). In the 1988 implementation this procedure is provided for by the communication application profile (T.521).

The optional functions can be negotiated at the beginning of an association and/or within an association between the interchange of different documents.

At the beginning of an association the 1984 implementation uses the S-CONNECT

service primitive. Between documents S-CAPAB-DATA is used. The "User Data" parameter of these primitives contains the information being negotiated. This information consists of protocol elements.

In case of the 1988 implementation the same session service primitives are used. However, these service primitives convey APDUs of the application layer. In section 5.2 it was shown that this does not cause any problems for the interworking between the two implementations.

Conclusion: The negotiation procedure does not show any incompatibilities between the 1984 and 1988 implementation.

References

[5], [11], [12], [17], [18]

Chapter 6

Conformance testing

6.1 Introduction

In chapter 2 the importance of conformance testing was pointed out. This chapter contains a description of the conformance testing methodology as defined by the ISO Conformance Testing Framework and Methodology (ISO 9646), see [19]. According to this standard, the primary purpose of conformance testing is to increase the probability that different implementations are able to interwork. This means that conformance testing, on its own, is not a sufficient condition to guarantee interworking capability. When two implementations conform to the same standard, they may fail to interwork because of factors outside the scope of the standard. Furthermore, conformance testing detects errors rather than their absence.

When does an implementation conform to a standard?
A real system is said to exhibit conformance to a standard, if it complies with the requirements of this standard during communication with other real systems. Those standards include protocol standards (single-layer and multi-layer) and transfer syntax standards. Conformance of a real system is expressed at two levels:

- Conformance to each individual standard, i.e. the standard for a layer in the open system.
- Conformance to the set of inter-related-standards, which together define the behaviour of the system during communication.

Exhaustive testing of these standards is because of the complexity of most protocols[1], both on technical and economical grounds, impractical. Therefore, four types of

[1] the number of possible combinations of events and timing of events is astronomical

testing are distinguished. These are discussed in section 6.3. First the conformance requirements, i.e. those requirements that are applicable according to the standard, are discussed in section 6.2. In section 6.4 abstract test methods and in section 6.5 abstract test suites and the notation used are described.

6.2 Conformance requirements

Conformance requirements are those requirements that, according to the standard, must be met by the system and which need to be tested. It is necessary that there is an unambiguous and objective understanding of the conformance requirements of a standard. They can be stated in a standard as:

- mandatory requirements, i.e. observed in all cases;
- conditional requirements, i.e. observed only if the cases mentioned in the standard apply;
- optional requirements, i.e. observed only when they are selected.

Except for this way of looking at conformance requirements, there are also two other ways.
First, conformance requirements can be stated as:

- positively, i.e. they state what shall be done;
- negatively. i.e. they state what shall not be done.

Secondly, conformance requirements can be grouped into:

- static conformance requirements (SCR);
- dynamic conformance requirements (DCR).

To avoid ambiguity, these static and dynamic groups should be stated separately from one another. At the moment however, most standards are not written in this way. When a system or implementation satisfies both the static and dynamic requirements, it is called a *Conforming System.* The SCR are discussed in section 6.2.1. The DCR are described in section 6.2.2. In section 6.2.3 and section 6.2.4 the documents, which specify what static conformance requirements are implemented, are discussed.

In the next discussion a distinction is made between a System Under Test (SUT) and an Implementation Under Test (IUT). An IUT is an implementation of one or

more protocols in an adjacent user/provider relationship, being part of a real open system (SUT) which is to be studied by testing.

6.2.1 Static conformance requirements

The static conformance requirements state the allowed minimum capabilities of an implementation, in order to facilitate interworking. They can be of two varieties:

1. those requirements which concern the capabilities to be included in the implementation;
2. those requirements which concern multi-layer dependencies, i.e. placing constrains on the underlying layers of the system.

6.2.2 Dynamic conformance requirements

Dynamic conformance requirements are all those requirements which determine what observable behaviour is permitted by the relevant standard during communication. They form the bulk of each standard and define the set of allowable behaviours of an implementation (usually by means of state tables in the standard). This set defines the maximum capability that a conforming implementation can have.

6.2.3 Protocol Implementation Conformance Statement

The protocol implementation conformance statement (PICS) specifies the capabilities and options which have been implemented. This PICS is necessary to evaluate the conformance of an implementation. During testing, only the requirements mentioned in the PICS are tested. This PICS is made by the manufacturer of the equipment to be tested, according to a PICS proforma, which is specified in the standard (e.g. in the form of an annex).

The information in the PICS can be of two categories:

- information related to the mandatory, conditional and optional static conformance requirements of the protocol itself;
- information related to the mandatory, conditional and optional static conformance requirements for multi-layer dependencies.

There is one PICS for each abstract test suite to be run against an IUT.

When a set of inter-related protocols is implemented, a **Systems Conformance Statement** is also necessary. This statement states all the protocols for which PICSs are provided.

6.2.4 Protocol Implementation eXtra Information for Testing

In addition to the information provided by the PICS, the test laboratory will require information relating to the implementation under test (IUT) and its testing environment. This extra information is provided by the protocol implementation extra information for testing (PIXIT), which again is provided by the manufacturer of the equipment to be tested. This is done in the same way as the PICS, i.e. according to a PIXIT proforma in for instance an annex of the standard.

The information contained by the PIXIT may be:

- information about the system under test (SUT), e.g. addressing information;
- information which adds precision to the information provided by the PICS;
- information to help determine which capabilities stated in the PICS are testable and which are untestable;
- other administrative information, e.g. IUT identification information.

There is one PIXIT for each abstract test suite to be run against an IUT, and the PIXIT should not conflict with the PICS.

6.3 Types of testing

As mentioned in the introduction of this chapter exhaustive testing is impractical. Four types of testing are distinguished, depending on the extend to which they give an indication of conformance. These types are:

- Basic interconnection tests,
- Capability tests,
- Behaviour tests and
- Conformance resolution tests.

These types of testing are discussed in the next subsections. In the last two subsections the test overview and the analysis of the test results are described.

6.3.1 Basic interconnection tests

To investigate whether an implementation provides sufficient conformance to make basic interworking possible, basic interconnection testing is used. It provides limited testing of an IUT.

These tests are appropriate:

- for detecting severe cases of non-conformance;
- to decide whether or not to run capability and behaviour tests;
- for checking addressing and other matters concerned with the test environment;
- to determine whether the implementations appear to be usable for communication with other conforming implementations.

These tests are inappropriate:

- as a basis for claims of conformance (when taken on their own);
- as a means to determine causes for communications failure.

They should be standardized as a very small test suite or as a subset of the conformance test suite.

6.3.2 Capability tests

Capability tests are used to test, in a limited way, the SCR. The tests are used to determine which capabilities stated in the PICS can be observed and to check that those observable capabilities are valid with respect to the SCR.

These tests are appropriate:

- to check that the capabilities of the IUT are consistent with the SCR;
- to check as far as possible the consistency of the PICS with the IUT;
- to enable efficient selection of behaviour tests;
- to check that the capabilities of the IUT are adequate to meet a particular user's need;
- when taken together with the behaviour tests, as a basis for claims of conformance.

These tests are inappropriate:

- on their own as a basis for claims of conformance;
- for testing in detail the behaviour associated with each capability;
- for solution of problems experienced during live usage.

Capability tests are standardized within a conformance test suite, and may be executed as a separate group or together with the behaviour tests.

6.3.3 Behaviour tests

To test an implementation as thoroughly as practical, behaviour tests are used. These tests cover the full range of the DCR. Since the tests are designed to be run collectively in a single test environment, it is possible that faults, which are difficult to detect, are likely to be missed. This means that it is possible that a non-conforming implementation passes the tests. The number of times this happens, must be minimized.

These tests are appropriate:

- as a basis for claims of conformance, when taken together with capability tests.

These tests are inappropriate:

- for resolution of problems experienced during life usage or where other tests indicate possible non-conformance even though the behaviour tests have been satisfied.

They are standardized as the main part of a conformance test suite.

6.3.4 Conformance resolution tests

To test a narrow field of the protocol, the conformance resolution tests are used. These tests provide diagnostic answers whether an implementation satisfies particular requirements. They are used to test aspects of the protocol which are untestable using standardized conformance tests.

These tests are appropriate:

- for providing a yes/no answer in a strictly confined and previously identified situation;
- for identifying and offering resolutions for deficiencies in a current conformance test suite.

These tests are inappropriate:

- as a basis for judging whether or not an implementation conforms overall.

The tests are not standardized.

6.3.5 Test procedure overview

The total test procedure is called the conformance assessment process. This process involves three phases:

1. Preparation for testing, which involves:
   (a) production of the System Conformance Statement, PICS and PIXIT;
   (b) choice of abstract test method and abstract test suite, based on those documents;
   (c) preparation of the SUT and real tester.
2. Test operations involving:
   (a) a static conformance review;
   (b) test selection and parameterization based on PICS and PIXIT;
   (c) one or more "test campaigns"
3. Test report production, described in section 6.3.6.

In figure 6.1 this process is illustrated.

Static conformance review contains analysis of the SCR and the PICS for consistency between the both and for the consistency of the PICS on its own.

During test selection the appropriate test cases of the conformance test suite are selected using both PICS and PIXIT. Then these test cases are parameterized using the PIXIT. Before or after this phase in the process, the abstract test cases are converted into executable test cases. The result of conversion, selection and parameterization is called a *parameterized executable test suite,* which is the actual test suite to be run.

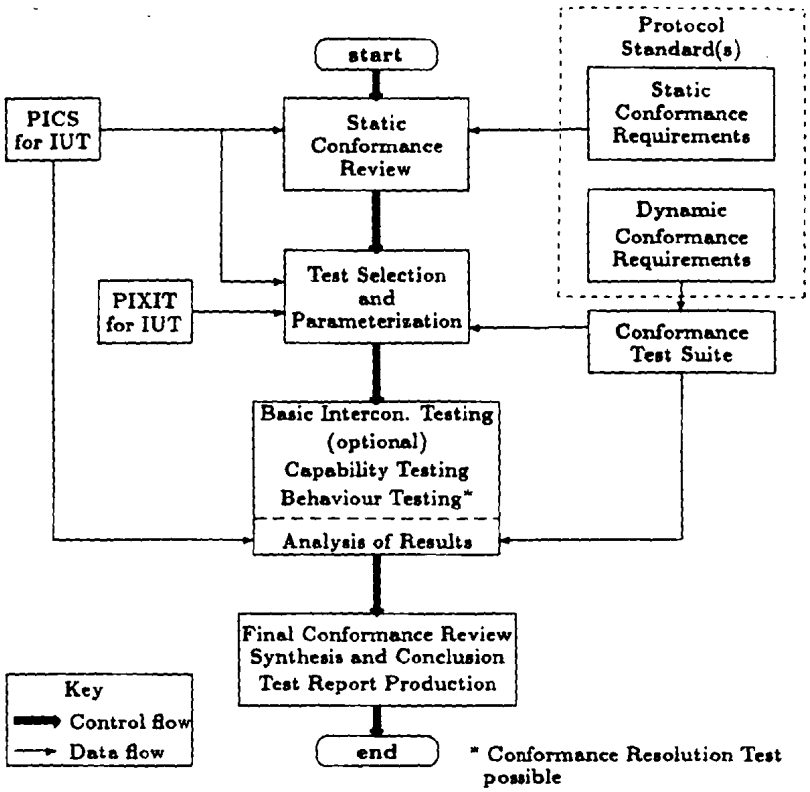


Figure 6.1: *Conformance assessment process overview.*

A test campaign is the process of executing the parameterized executable test suite and includes the following three testing types:

- basic interconnection testing;
- capability testing;
- behaviour testing.

6.3.6 Analysis of test results

In this section the phases "Analysis of Results", "Final Conformance review", "Synthesis and Conclusions" and "Test Report Production" mentioned in figure 6.1 are discussed.

The analysis of the results may be interleaved with the execution of the different groups of test cases. However, it is easier to think of it coming after the test execution step. To analyze the results, distinctions has to be made between observed outcomes and foreseen outcomes. Observed outcome is the series of events which occurred

during test execution. Foreseen outcomes are defined primarily in abstract terms (there can be more than one foreseen outcome). With every foreseen outcome a verdict is associated. A verdict is a statement of:

- pass, i.e. the observed test outcome gives evidence of conformance;
- fail, i.e. the observed test outcome either demonstrates non- conformance or contains at least one syntactical invalid or inopportune test event;
- inconclusive, i.e. neither a pass nor a fail verdict can be given.

Analyzing the results means comparing the observed outcomes with the foreseen outcomes. The test verdict assigned to the observed outcome is that associated with the matching foreseen outcome. If the observed outcome was unforseen, then the abstract test suite specification will state what default verdict shall be assigned.

The final conformance review involves synthesis of the results of the behaviour tests, with what has been learned from the analysis of the PICS and the results of the capability tests.

With the result of this synthesis it can be seen whether the test results are consistent with the capabilities stated in the PICS and conclusions on the conformance can be obtained.

During the test report production phase, the results of the conformance testing are documented in conformance test reports. There are two types of reports:

- System Conformance Test Report (SCTR), which is always provided and gives an overall summary of the conformance status of the SUT;
- Protocol Conformance Test Report (PCTR), which is specified for each protocol tested in the SUT and documents all the results of the test cases.

If there are no fail verdicts in the PCTR, then the client of the test laboratory can claim that the IUT is a conforming implementation.

Note: To achieve the object of credible conformance testing it is clear that whenever performing the test, the results must be the same. However experience has shown that this is not always possible and therefore every effort must be made to minimize the probability that this happens.

6.4 Abstract test methods

Since real systems can come in a wide variety of configurations and vary in the ways in which their behaviour can be observed and controlled during testing, a range of test methods is designed. The following three basic configurations of SUTs can be identified:

Configuration 1: 7-layer open system, i.e. protocols in all 7 layers;

Configuration 2: Partial (N)-open system, i.e. protocols in layers 1 to N;

Configuration 3: Open relay-system, i.e. Network relay-system (layer 1 to 3) or Application relay-system (layer 1 to 7).

An IUT can be defined for configurations 1 and 2 as a *single-layer IUT* or as a *multi-layer IUT*. An IUT of configuration 3 will include at least the layer which provides the relay function.

An abstract test method is described by identifying the points closest to the IUT, at which control and observation are to be exercised.

In the next section these points of control and observation (PCOs) are described. In section 6.4.2, the abstract testing functions are discussed. Finally, an overview of the abstract test methods is given in section 6.4.3.

6.4.1 Points of Control and Observation

The behaviour of a protocol entity (i.e. the DCR) is defined in terms of PDUs and Abstract Service Primitives (ASPs). The behaviour of a N-entity is defined by N-ASPs and (N-1)-ASPs (including N-PDUs), see figure 6.2. The behaviour of a multi-layer IUT is defined by the ASPs above and below that IUT, including the PDUs of the protocols in the IUT.

Possible PCOs are identified by three factors:

1. whether it is the ASPs or the PDUs which are observed and controlled;
2. the layer identity of the ASPs or PDUs concerned;
3. whether they are controlled and observed within the SUT or in a system remote from the SUT.

The PCO below an IUT can be in the SUT or externally, i.e. in the tester. The latter possibility is assumed to be always possible. So, in case the ASP below the

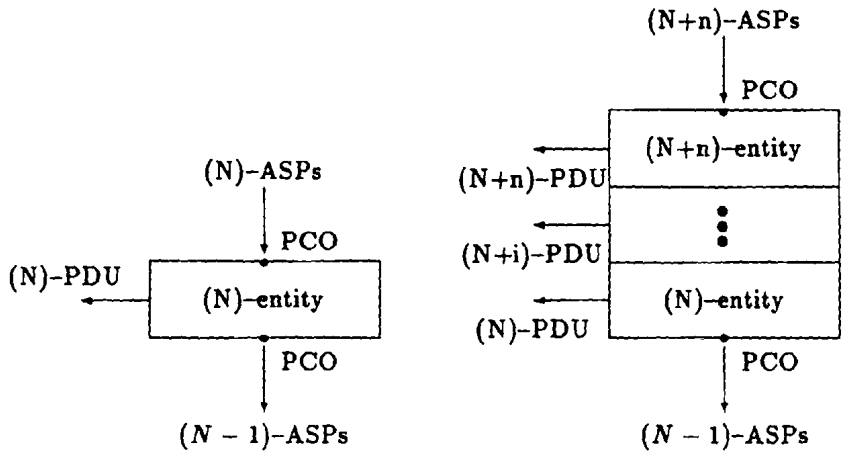


Figure 6.2: *Points of Control and Observation.*

IUT can not be controlled and observed locally, conformance testing can be carried out externally. In case the ASP above the IUT is not controllable nor observable, it is said to be hidden and not testable.

A PCO can be modelled as two queues:

- one output queue for control of test events to be send to the IUT;
- one input queue for observation of test events received from the IUT.

6.4.2 Abstract testing functions

In the previous section it was seen that the PCOs were distributed over the ASPs above the IUT and the ASPs (including PDUs) below the IUT. This means that two abstract testing functions can be distinguished:

Lower tester (LT): this is the testing function related to the control and observation of the lower boundary of the IUT. There are two situations:

- local, i.e. the tester is located in the lower part of the SUT;
- external, i.e. the lower tester relies on the (N–1)-service, provided by the lower tester itself, the SUT and the physical connection between the two.

Upper tester (UT): this is the testing function related to the control and observation of the upper boundary of the IUT. Not all events or states defined in the standard can be controlled and observed by an upper tester which can only

talk in terms of ASPs. Therefore the concept of *abstract local primitive* (ALP) is introduced. This is an abbreviation for a description of the control and/or observation to be performed by the upper tester, which can not be expressed in terms of ASPs. These ALPs relate to the events or states defined within the relevant standard.

The coordination and synchronization between the upper and lower tester is provided for by the *test coordination procedures (TCP)*. These test coordination procedures may be specified in various ways. Sometimes it is possible to use **a test management protocol (TMP)** to provide the coordination between the two testers. In other cases it is not possible to specify any coordination between the two.

6.4.3 Overview of abstract test methods

For IUTs defined within SUTs of configuration 1 and 2 (i.e. end-systems), four categories of abstract test methods are defined:

- local, i.e. observation and control is specified in terms of events occurring within the SUT;
- external:
  - distributed, i.e. the lower tester is situated remotely and the requirements to be met by the test coordination procedure are defined in the abstract conformance test suite (the procedure themselves are not defined);
  - coordinated, i.e. enhanced version of distributed method that uses a standardized upper tester and a test management protocol;
  - remote, i.e. the lower tester is situated remotely and in addition ALPs may be used.

Local test methods are only applicable to in-house testing by suppliers, because these test methods are characterized by observation and control specified in terms of events occurring within the SUT. External test methods are also applicable to in-house testing. Furthermore, they are applicable to testing carried out by users and third party test centers.

Test methods can also be distinguished depending on the number of layers they have to test. There are three variants:

- single-layer IUT;
- multi-layer IUT;

- embedded, i.e. one layer within a multi-layer IUT.

The preferred way of testing a multi-layer IUT is by incremental use of embedded methods, starting at the lowest layer and working up to the top.

For IUTs defined within SUTs of configuration 3 (i.e. relay-systems), two test methods are defined:

- loop-back, used for testing a relay-system from one subnetwork;
- transverse, used for testing a relay-system from two subnetworks.

In Appendix D the various test methods are illustrated and a summary is given of the terms, which specify the test events, for the various test methods (see also figure 6.2)

6.5 Abstract test suites

In this section the structure of test suites is discussed. A test suite contains the total set of test cases to test a protocol. These test cases are discussed in section 6.5.2. The process of producing these test suites is described in section 6.5.3 and the notation used to write the test cases, i.e. Tree and Tabular Combined Notation (TTCN), is shortly discussed in the last subsection.

6.5.1 Structure

Test suites have a hierarchical structure, consisting of:

- test groups;
- test cases;
- test steps;
- test events.

Test groups and test steps can be nested to an arbitrary depth. Figure 6.3 shows the test suite structure.

Test groups provide for a logical ordering of test cases. They may be used to aid planning, development, understanding or execution of test suites. A *test group*

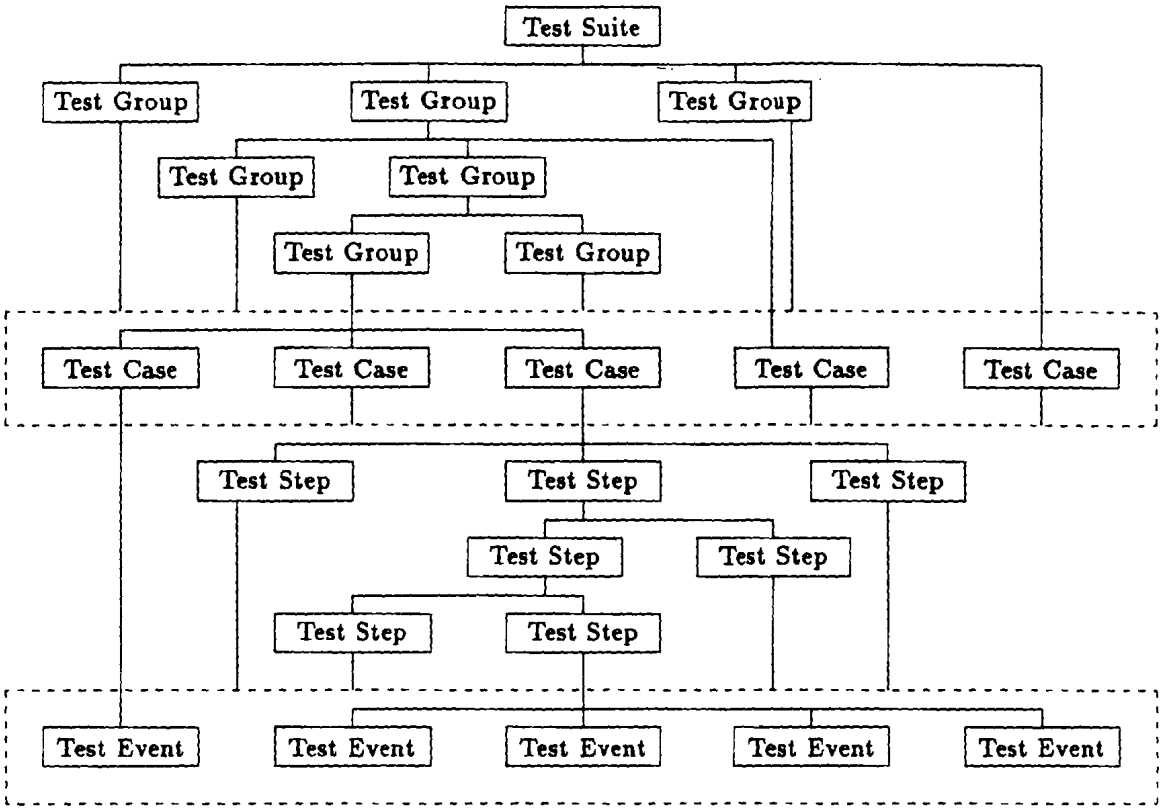


Figure 6.3: *Possible test suite structure.*

objective may be associated with a test group. The complete test group objective associated with a test group of a lower level is formed by concatenating the given test group objective with the higher level test group objectives.

The test case is an important level in the test suite. Each test case normally has a single test purpose. The complete test purpose is formed by concatenating the appropriate complete test group objective with the individual test purpose.

Test steps are used to modularize the test cases. Each test case comprises at least one test step (test body). Except for the test body a test case may include a series of test steps to put the IUT into a state required for the test body to start from (the preamble) or to put it into a stable state to end the test (the postamble).

Test events are indivisible units within a test step. Examples of test events are e.g. transfer of a single PDU or ASP to or from the IUT.

To assure an adequate coverage of the relevant conformance requirements the test suite must include test cases for whichever of the following categories is relevant:

- capability tests;

- behaviour tests of valid behaviour;
- behaviour tests of syntactically invalid behaviour;
- behaviour tests of inopportune behaviour, i.e. syntactically valid events occurring at the wrong time;
- tests focussed on PDUs send to the IUT;
- tests focussed on PDUs received from the IUT;
- tests focussed on interactions between PDUs sent and PDUs received;
- tests related to each mandatory capability;
- tests related to each protocol phase;
- variations in test events occurring in a particular state;
- timing and timer variations;
- PDU encoding variations;
- variations in values of individual parameters;
- variations in combinations of parameter values.

6.5.2 Test cases

Three types[2] of test cases can be distinguished:

- generic;
- abstract;
- executable.

A generic test case is one which:

1. provides a refinement of the test purpose;
2. specifies the sequence of test events in the test body which corresponds to verdicts of 'pass', 'fail' or 'inconclusive';
3. is used as the common root of corresponding abstract test cases for different abstract test suites;
4. includes a description of the initial state in which the test body should start (not specifying a preamble);

[2]In OSI 9646 only the abstract and executable test cases are distinguished. In the article OSI Conformance Testing of D. Rayner, also generic test cases are distinguished.

5. need not describe a postamble.

An abstract test case is derived from a generic test case and the relevant protocol standard. An abstract test case:

1. specifies the test case in terms of a particular test method;
2. adds a more precise specification for sequences of events;
3. adds the sequences of events composing the pre- and postamble.

An executable test case is derived from the abstract test case and is in a form that allows it to be run on a real tester.

6.5.3 Test suite production

A general test suite production process for specifying an abstract test suite does not exist. There are however some steps in the process of producing an abstract test suite, which must be included in the test suite design. These steps are mentioned below. The order mentioned, is the normal form of the process of test suite production. This order is not mandatory as long as the steps mentioned are included in the process.

The normal abstract test suite production process has the following steps:

1. study the relevant standards to determine the conformance requirements to be tested and what needs to be stated in the PICS;
2. determine the test groups which are needed to achieve an adequate coverage of the conformance requirements and refine the test groups into sets of test purposes;
3. specify generic test cases for each test purpose;
4. choose the test method and decide what restriction need to be placed on the assumed capabilities of the tester and test coordination procedures;
5. choose a test notation and in it specify the abstract test cases;
6. specify inter-relationships among the test cases and the PICS, in order to determine the restrictions on the selection of the test cases for execution and on the order in which they can be executed;
7. consider the procedures for maintaining the abstract test suite.

6.5.4 Tree and Tabular Combined Notation

The Tree and Tabular Combined Notation (TTCN) is described in [19, part 3]. However at RNL a slightly different version of this notation is used, RNL-TTCN. RNL-TTCN is developed because TTCN, which is not stable yet, shows some ambiguities. On the other hand RNL-TTCN is also still under development. TTCN is used to specify generic and abstract test suites for OSI/CCITT standards. It was designed to meet the following objectives:

- to provide a notation in which generic and abstract test cases can be expressed;
- to provide a notation which is independent of the test method used;
- to provide a notation which reflects the abstract testing methodology.

TTCN assumes that the basic test events are ASPs and timers. Abstract test cases can also be expressed in terms of PDUs.

TTCN is provided in two forms:

- TTCN.GR; i.e. graphical form which is suitable for human readability;
- TTCN.MP[3]; i.e. machine processable which is suitable for transmission of TTCN descriptions between machines and possibly suitable for other automated processing.

The two forms only differ in syntax; keywords instead of boxes are used as information delimiters.

In the next description of the test suite, RNL-TTCN is used.

```
R001  :  Suite  ::=  '$Suite'      Identifier
                                   DeclarationsPart
                                   ConstrainsPart
                                   PDUCodingRules
                                   DynamicPart
                     '$EndSuite'
```

In the *PDUCodingRules* all the parameter values of the PDUs are specified. This is only done for the parameters whose values are constant for the whole test suite.
In the *DeclarationsPart* the set of test events and all components used in the test suite are declared. Here also fields of the PDUs are declared, but not specified, i.e. they are given no value.

[3]RNL-TTCN is an extended subset of TTCN.MP.

It is necessary to specify the values of ASP parameters and PDU fields (not specified in the PDUCodingRules). This is done in the *ConstrainsPart*, where the encoding of these parameters is specified for every test case (or test step). In the *DynamicPart* finally the test steps, which make up the test cases, are described.

Note: Test cases could also be specified in Formal Description Techniques (FDTs), such as LOTOS (based on Calculus of Communicating Systems) or SDL (based on extended finite state machine model). However these FDTs were developed to specify protocols and services. TTCN, on the other hand, was developed to specify conformance test suites, resulting in some advantages of TTCN above FDTs, see [23]. In future the developments tend to converge to the use of formal techniques of some kind.

References

[19], [20], [21], [22], [23], [24]

Chapter 7

Test suite

7.1 Introduction

This chapter describes the testing methodology which will be used for the testing of facsimile group 4 layer 7. This description of the testing methodology includes the choice of the test method and a description of the (functional) architecture of the protocol tester. These subjects are discussed in section 7.2. In section 7.3 the problems which occurred during the derivation of the test cases are given. These test cases are written in RNL-TTCN and given in Appendix E. In the last section the results of the RNL-TTCN-parser are given.

7.2 The test method

In this section, the test method which will be used and the arguments for its choice are given. At the end of this section the architecture of the tester is described.

7.2.1 Choice of the test method

Before the test method and the arguments for its choice are given, the five different types of test methods are recollected. The following test methods can be distinguished:

- local test methods,
- coordinated test methods,

- distributed test methods,
- remote test methods and
- relay test methods.

After one of these test methods is chosen, it has to be determined whether a single-layer, multi-layer or embedded method is used.

There are four arguments which have to be considered, before choosing a specific test method. These arguments are:

1. relay-system or end-system;
2. what is the subject of the test;
3. the accessibility of the SAPs;
4. the possibility to implement a standardized TMP.

Since testing of facsimile group 4 means testing of an end-system (ISDN telematic terminal), the relay test methods are not applicable.

Considering the second argument, a local test method will not be appropriate, since the subject of testing is a complete ISDN telematic terminal (no prototype). This implies that the IUT can not be isolated and one of the external test methods has to be chosen.

Considering the third argument, a distributed test method is not applicable since the distributed test method needs a SAP above the IUT which is accessible. This is in case of an ISDN telematic terminal, where the upper boundary is controlled via a keyboard and paper inlet and outlet, not very probable. For the same reasons the Test Coordination Procedures can only be informal descriptions of the actions of a human operator.

Considering the last argument leaves the remote test method as the method to be used, since it will probably not be possible to implement a standardized TMP in an upper tester. Furthermore, the remote testmethod is the most practical one since it can be applied universally (testing at a distance over a network is possible).

Now that the remote test method is chosen, it has to be determined which form of remote testing will be used. The fact that ISDN telematic terminals are always multi-layer implementations makes single-layer testing not applicable. This means a choice has to be made between multi-layer testing or single-layer embedded testing. Chosen is for a single-layer embedded test method (RSE), because this method opens the way for a modular approach for the tests and the inter-layer boundaries can be controlled and observed.

Note: For the uppermost layer (i.e. layer 7) the remote single-layer embedded test method is equal to the remote single-layer test method. Since tests are derived for layer 7, in this document there will not be any difference between the two methods. However, when testing lower layers this is not true any more.

The conclusion can be that the remote single-layer test method will be used in an incremental way to test a facsimile group 4 terminal. In Figure 7.1 the chosen method is illustrated.

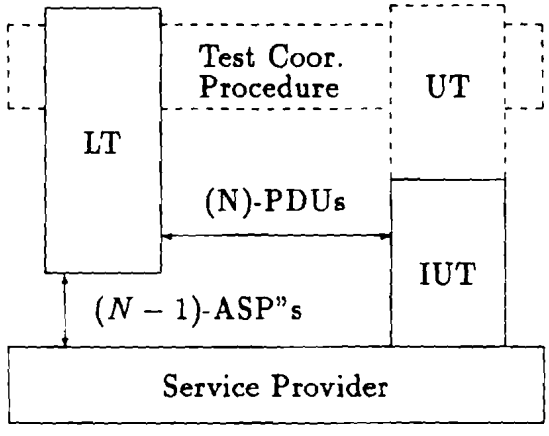


Figure 7.1: *The RSE test method.*

7.2.2 Functional architecture of the testsystem

In the previous subsection the RSE test method was mentioned as the method to be used by the CTS-B project. This means that a complete SUT (= ISDN telematic terminal) will be tested layer by layer in an upward direction, starting at layer 1. Since tests for layer 1 are not developed within the project (these tests will be bought), it is assumed that layer 1 is already tested successfully.

The ASPs at each layer boundary must be observable and controllable in the test-system. This leads to the functional architecture of the testsystem as shown in figure 7.2. At each layer, the ASPs are used for control of that layer from the layer above or from a test suite to test the adjacent layer of the peer entity.

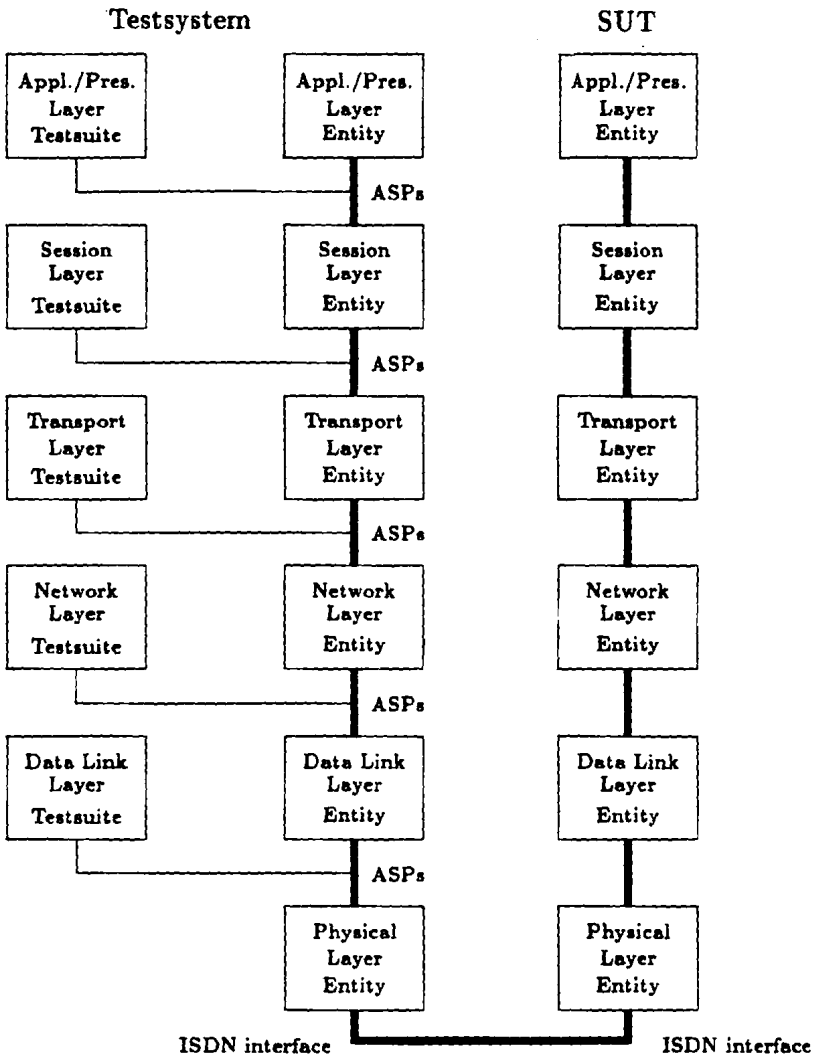


Figure 7.2: *Functional architecture of the testsystem.*

7.2.3 The reference implementation

Up to now the reference implementation, see chapter 2, has not been discussed. However it plays an important role in the testsystem development. In order to guarantee the quality of the testsystem and the harmonization of all testsystems, it is necessary to test the testsystem against a reference implementation. This will be done by connecting the testsystem to the reference implementation, which plays the role of the SUT. The testreports of this test must be identical to those of tests performed by other testsystems against the same reference implementation.

A possible reference implementation can be the protocol entities within the testsystem. In that case two identical testsystems will be connected to each other. One of these testsystems is used as the reference implementation, the other one executes

the test suites.

In case of the CTS-B project a reference implementation of FAX 4 layer 7 will be implemented on a Personal Computer using C–language. The lower layers, i.e. layers 1 to 5, will be handled by an ISDN cart in the PC. The reference implementations of the other layers are made by the other partners in the CTS-2-ISDN project.

7.3 Test suite production

In this section the test suite, which will be used to test the compatibility of 1988 implementation with a 1984 implementation, is derived. Before this can be done, first the test suite structure, the test method and the test notation have to be determined. The latter two have already been chosen.

- Test method = single-layer embedded remote test method (RSE);
- Test notation = Tree and Tabular Combined Notation (TTCN).

This means that only the structure has to be determined. This is done in section 7.3.1. The test suite itself is given in Appendix E.

7.3.1 Test suite structure

The title of this section is "Test *suite* production". But why a test suite? Another possibility could be to put all the test cases into one testgroup, which belongs to the test suite for testing FAX 4 in general. However, new FAX 4 terminals will be implemented according to the 1988 recommendations, meaning that at the moment there is a period of overlap. But in future all terminals will work according to the same recommendation (or perhaps new recommendations). This makes the test cases not applicable any more (in case of new recommendations, other tests might be necessary). When part of the general test suite for FAX 4, it means that this test suite has to be changed. Furthermore, a customer of the test-laboratory might not want his product to be tested on compatibility with the older recommendations. So a test suite seems the best solution.

The purpose of the test suite is to test the compatibility. Therefore, the test suite can be, depending on the various incompatibilities, split into test cases. The incompatibilities encountered are summarized below.

1. different use of the S-U-ABORT service;

2. different use of the S-ACT-RESUME service[1];
3. different use of the attribute "Reference to subordinates"[2];
4. different use of the attribute "Number of discarded pels"[2];
5. use of the attributes "Content architecture class" and "Interchange format" according to ETSI;
6. different coding of the attribute "Pel transmission density" and
7. different coding of the attribute "compression".

This means that seven test cases can be distinguished. The purpose of these test cases is to show whether or not two implementations according to the different standards can interwork. This introduces a problem in the use of the Conformance testing methodology. Instead of an implementation which has to be tested on its conformance to a standard, now interworking between two different standards with the same functionality has to be tested.

The problem can be clarified by the next example. Suppose the 1984 implementation sends a PDU which is, according to the 1988 implementation, coded incorrectly. The 1988 implementation will react with an abort procedure. This is according to the 1988 standard correct, and therefore a PASS verdict would be appropriate. On the other hand, since the connection is terminated, the verdict about the compatibility must be FAIL.

To give a verdict about the compatibility, one could change the definition of the verdicts PASS and FAIL. A verdict PASS will be assigned to a test, when this test shows that the implementation is compatible. A verdict FAIL will be assigned in the other case. An important reason for not using this solution is the fact that by changing these definitions, one also deviates from the conformance testing methodology and framework, [19].

The problem is solved by using a boolean variable (*compat*, see Appendix E) indicating whether the 1988 implementation is compatible (value of compat is true) or not (value of compat is false) with the 1984 implementation. The value of this variable has to be reported to the tester. During the "Analysis of Results" phase, see Figure 6.1, only the foreseen and observed outcomes of the verdicts are compared, meaning that other variables are not reported. Therefore an extra mechanism is needed to report the value of the variable *compat* to the tester. This can be done in RNL-TTCN by using an extra PCO to save this value on a predefined memory location, see Figure 7.3. This extra PCO is called *OUT* in Appendix E.

[1]The 1988 recommendations do not specify this service. The 1984 recommendations on the other hand do specify the service. However, at the moment facsimile terminals, as far as known, do not have the possibility to resume transfer.

[2]At the moment this attribute is not used by FAX4 terminals available on the market

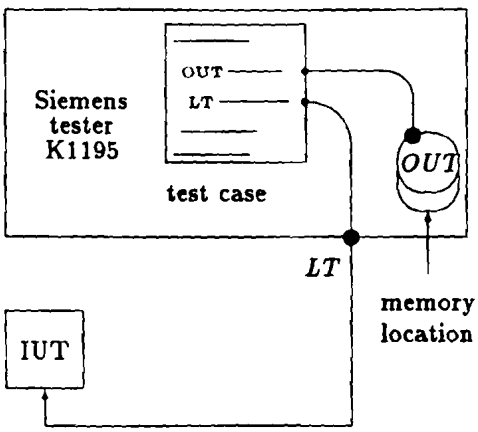


Figure 7.3: *Schematic rendition of PCOs.*

Another problem still remains: How to define this extra PCO in the compiler (see Figure 7.4). This compiler is used to generate FORTH code (used by the Siemens K1195 protocol analyzer) out of a RNL-TTCN test suite. At the moment it is not sure how to solve this problem. Since it depends on what is requested by the test suite and what is possible to implement in the compiler, consultation with the implementor of the compiler is necessary.

7.4 Results of the parser

The test suite given in Appendix E has been parsed by a RNL-TTCN parser, and is free of syntax errors. Since this parser was still under development at the moment of graduation, the test suite given is parsed in two parts. These two parsers are (see Figure 7.4):

- parser for Declaration Part (partly), Constraints Part and Coding Part (concheck);
- parser for Declaration Part (partly) and Dynamic Part (dyncheck).

A curved vector in Figure 7.4 indicates that it is still vague how to implement these "connections".

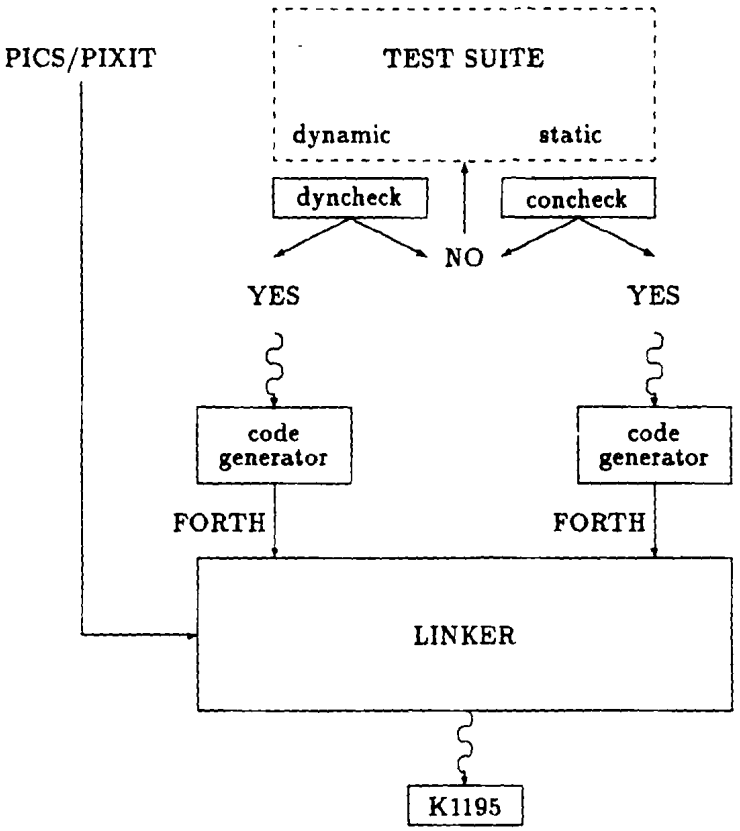


Figure 7.4: *Functional units of compiler.*

References

[19], [20], [21], [26]

Chapter 8

Conclusions

In this report the interworking, between facsimile group 4 apparatus implemented according to the 1984 recommendations and implemented according to the 1988 recommendations, was described. Altogether it can be stated that the 1988 recommendations were designed to be compatible with the 1984 recommendations. There are however some differences which may cause incompatibilities.

Both, communication aspects and document aspects, may introduce incompatibilities between the two implementations. These incompatibilities are:

- The use of a confirmed S-U-ABORT service by the 1984 implementation, and the use of an unconfirmed S-U-ABORT service by the 1988 implementation.
- The use of a resume procedure by the 1984 implementation may cause an incompatibility, since the 1988 implementation is not able to resume from an error occurring during document transmission.
- The use of the attribute "Reference to Subordinate Object" by the 1984 implementation. Its counterpart in the T.400/T.500 series is the attribute "Subordinates", which is not used by the 1988 implementation.
- The attribute "Number of discarded pels" can have any value in the 1984 implementation, but has restricted values in the 1988 implementation.
- The attributes "content architecture class" and "interchange format class" are not used according to the document application profile. According to the document profile, see [11, T.414 Annex B], and ETSI, see [18], they are mandatory. However, when used, they introduce an incompatibility with the 1984 implementation
- The different coding of the attributes "pel transmission density" and "compression" used during the capability negotiation.

To test this compatibility, a test suite in RNL–TTCN is derived. Since interworking is tested, an extra point of control and observation is used to report the result of the compatibility test. At the moment it is still unclear how this must be implemented. Further consultation with the implementor of the compiler is necessary to specify this point of control and observation.

Since the RNL-TTCN parser was still under development, the test suite derived was parsed in two parts:

- the static part, i.e. declarations, constraints and coding rules, and
- the dynamic part.

Both parts are free of syntax errors.

The above shows that some work still has to be done. This can be summarized in the next items:

- The complete test suite has to be parsed, when the complete parser is implemented.
- During the implementation of the compiler, consultation is needed with the implementor of the compiler.
- In the final test system, the semantics of the test suite have to be checked and corrected in case of errors.
- The incompatibilities found, must be reported to CCITT.
- The problems encountered during the study of the various recommendations and summarized in Appendix F must be reported to CCITT.

Appendix A

Summary of FAX 4

A.1 Summary of the attributes

In the Tables A.1 until A.4 the mandatory (m), non-mandatory(nm) and the default (d) attributes of the various descriptors and text units are given. Distinction is made between the 1984 and 1988 recommendation. In Table A.5 the options which are not necessarily implemented and therefore subject of a negotiation during session establishment phase are given.

1984 recommendation		
Attributes	Class	Defined Values
Presentation capabilities:		
Basic terminal characteristics	m	Facsimile
Interchange format	m	TIF0
Non-basic terminal capabilities	NM	see table A.5
1988 recommendation		
Attribute	Class	Permissable values
Document profile descriptor	M	
Specific layout structure	m	present
Document characteristics	M	
Document application profile	m	FAX 4
Document architecture class	m	FDA
Non-basic document characteristics	NM	
Page dimensions	nm	N. Am. = 10200,13200 ISO B4 = 11811,16677 ISO A3 = 14030,19840 Jap. Leg. = 12141,17196 Jap. Let. = 8598,12141 (see note 1)
Raster-graphics coding attributes	NM	
Compression	nm	uncompressed
Raster-graphics presentation features	NM	
Pel transmission density	nm	5 BMU (240 ppi) 4 BMU (300 ppi) 3 BMU (400 ppi)

Table A.1: *Attributes of the presentation capabilities descriptor and document profile.*

1984 recommendation				
Attributes	Class doc/page	Basic value	Default value	Non-basic value
Object type	m/m	Document/ page		
Object identifier	nm/nm			
Ref. to subordinate object	nm/-	page identifier	none	none
Ref. to content portions	-/nm	content portion id.	none	none
Default value list	nm/-	page		
Presentation attributes	-/d			
Dimensions	-/d	*hor* $\leq$ 9920 *ver* $\leq$ 14030	*hor* $\leq$ 9920 *ver* $\leq$ 14030	see table A.5

1988 recommendation				
Attribute	Class root/page	Basic value	Default value	Non-basic value
Shared				
Object type	m/m	doc layout root/ page	none	none
Object identifier	nm/nm	note 2	none	none
Content portion	-/nm	note 3	none	none
Default value list	nm/-	note 4	none	none
Layout				
Presentation attributes	-/d	see table A.3		
Dimensions	-/d	hor = 9920 BMU ver = 14030	hor = 9920 BMU ver = 14030	N.Am. ISO B4 ISO A3 Jap. legal Jap. Letter

Table A.2: *Attributes of specific layout descriptors.*

1984 recommendation				
Attributes	Class	Basic value	Default value	Non-basic value
Content type	d	photographic	photographic	none
Pel path	d	0^0	0^0	none
Line progression	d	270^0	270^0	none
Pel transmission density	d	200	200	see table A.5
1988 recommendation				
Attribute	Class	Basic value	Default value	Non-basic value
Content type	d	Raster-graphic (T.417)	Raster-graphic (T.417)	none
Pel path	d	0^0	0^0	none
Line progression	d	270^0	270^0	none
Pel transmission density	d	6 BMU (200 ppi)	6 BMU	5,4,3 BMU

Table A.3: *Presentation attributes.*

1984 recommendation				
Attributes	Class	Basic value	Default value	Non-basic value
Content portion id.	nm		none	none
Type of coding	d	T.6	T.6	none
Coding attributes for T.6 photographic information:				
number of pels per line	d	any number	note 5	
Number of discarded pels	d	any number		
Compression	d	as in T.6	as in T.6	as in T.6
1988 recommendation				
Attribute	Class	Basic value	Default value	Non-basic value
Content info	m	T.6 string	none	none
Content portion id.	nm	note 3	none	none
Type of coding	d	T.6	T.6	none
Raster-graphic coding attributes				
Number of pels per line	d	1728 note 6	1728	none
Compression	d	compressed	compressed	uncompressed
Number of discarded pels	d	37 note 7	37	none

Table A.4: *Attributes of text units.*

1984 recommendation		
Attributes	Class	Values for negotiable facilities
Non-basic terminal cap.	nm	
Page descriptor		
Dimensions	d	10200x13200 (N.A.), 11811x16677 (ISO B4), 14030x19840 (ISO A3)
T.6 Coding attributes		
Compression	d	uncompressed as in T.6
Pel transmission density	m	240, 300, 400 ppi

Table A.5: *Summary of the negotiable attributes.*

Notes:

1. The dimension attribute is represented by a data element that consists of two integers, specifying width and height in BMU. An indefinite page length is represented by a variable measure in vertical dimensions. In that case the value of this data is arbitrary and should be the nominal page length.
2. The object identifiers of the specific layout object descriptions are composed of sequences of numbers, each of these numbers represents a particular level of the specific layout structure. The number assigned to the specific layout root description is "1". The subordinate pages have a second number which uniquely identifies a particular page
3. Content portion identifiers are composed of the page identifier (see note 2) to which the content portion belongs and an additional number which identifies the content portion. The value of the attribute "content portions" consists of the last number of the content portion identifier.
4. In the default value list, specified in the root, default values for the dimensions and presentation attributes of the page can be specified.
5. the default number of pels per line is determined according to table 3.1
6. In case of ISO A4 the basic/default value is 1728. For the other case see Table 3/T563 of recommendation T.563.
7. In case of ISO A4 the basic/default value is 37. For the other case see Table 3/T563 of recommendation T.563.

A.2 ASN.1 description of the PDUs

ProtocolElement	::= CHOICE {
specificLayoutDescriptor	[2]IMPLICIT LayoutDescriptor,
textUnit	[3]IMPLICIT Text-Unit
presentationCapabilitiesDescriptor	[4]IMPLICIT PresentationCapabilities }

Note: A document profile is not specified for TIF0, ODIF class B specifies a profile

Table A.6: *Formal definition of protocol element.*

PresentationCapabilities	::= SET {
basicTerminalCharacteristics	[0]IMPLICIT BasicTerminalCharacteristics,
interchangeFormat	[1]IMPLICIT InterchangeFormat,
nonBasicTerminalCapabilities	[2]IMPLICIT NonBasicTerminalCapabilities OPTIONAL}
BasicTerminalCharacteristics	::= OCTET STRING - - 2 indicates facsimile according to T.5 and T.6
InterchangeFormat	::= OCTET STRING - - 0 indicates TIF0
NonBasicTerminalCapabilities	::= SET {
pageDimensions	[2]IMPLICIT SEQUENCE OF MeasurePair OPTIONAL,
codingAttributes	[3]IMPLICIT SEQUENCE OF CodingAttributes OPTIONAL,
presentationAttributes	[4]IMPLICIT SEQUENCE OF PresentationAttributes OPTIONAL}
MeasurePair	::= SEQUENCE{Measure,Measure}
Measure	::= CHOICE {
fixedMeasure	[0]IMPLICIT INTEGER,
variableMeasure	[1]IMPLICIT INTEGER}
CodingAttributes	::= CHOICE{
compression	[0]IMPLICIT Compression}
Compression	::= INTEGER{ uncompressed (0), compressed (1)}
PresentationAttributes	::= CHOICE{
pelTransmissionDensity	[11]IMPLICIT PelTransmissionDensity OPTIONAL}
PelTransmissionDensity	::= INTEGER{p200 (1), p240 (2), p300 (3), p400 (4)}

Note: In the 1988 recommendations the tag of compression = [2], and the tag of pelTransmissionDensity = [2]
The 1988 recommendation must specify according to ETSI: content-architecture-classes and interchange-format-class

Table A.7: *Formal definition of presentation capabilities.*

```
LayoutDescriptor                ::= SEQUENCE {
   layoutObjectType             LayoutObjectType,
   layoutDescriptorBody         LayoutDescriptorBody OPTIONAL}

LayoutObjectType                ::= INTEGER {
                                document (0), page (2) }

LayoutDescriptorBody            ::= SET {
   objectIdentifier             ObjectReferenceName OPTIONAL,
   referencesTo                 ::= CHOICE {
     subordinateObjects         [0]IMPLICIT SEQUENCE OF NumericString,
                                  - only used by document descriptor
     contentPortions            [1]IMPLICIT SEQUENCE OF NumericString}
                                  OPTIONAL,
   dimensions                   [4]IMPLICIT MeasurePair OPTIONAL,
   presentationAttributes       [6]IMPLICIT PresentationAttributes OPTIONAL,
   defaultValueList             [7]IMPLICIT SEQUENCE OF DefaultValueList
                                  OPTIONAL}

ObjectReferenceName             ::= [APPLICATION 1]IMPLICIT PrintableString

DefaultValueList                ::= CHOICE{
   pageAttributes               [2]IMPLICIT PageAttributes}

PageAttributes                  ::= SET {
   dimensions                   < Attribute OPTIONAL,
   presentationAttributes       < Attribute OPTIONAL }

Attribute                       ::= CHOICE {
   dimensions                   [1]IMPLICIT MeasurePair,
   presentationAttributes       [3]IMPLICIT PresentationAttributes}
```

Note: referencesTo subordinateObjects is not specified in the 1988 recommendations

Table A.8: *Formal definition of layout descriptor.*

```
PresentationAttributes          ::= SET {
   contentType                  ContentType OPTIONAL,
   photographicAttributes       [1]IMPLICIT PhotographicAttributes
                                  OPTIONAL}

ContentType                     ::= [APPLICATION 2]IMPLICIT
                                  INTEGER {photographic (1)}

PhotographicAttributes          ::= SET {
   pelPath                      [0]IMPLICIT OneOfFourAngles OPTIONAL,
   lineProgression              [1]IMPLICIT OneOfTwoAngles OPTIONAL,
   pelTransmissionDensity       [2]IMPLICIT PelTransmissionDensity
                                  OPTIONAL}

OneOfFourAngles                 ::= INTEGER {d0 (0)}

OneOfTwoAngles                  ::= INTEGER {d270 (3)}
```

Table A.9: *Formal definition of presentation attributes.*

```
TextUnit                          ::= SEQUENCE{
   contentPortionAttributes       Content-Portion-Attributes OPTIONAL
   textInformation                TextInformation }

ContentPortionAttributes          ::= SET {
   contentPortionIdentifier       PortionReferenceName OPTIONAL,
   typeOfCoding                   TypeOfCoding OPTIONAL,
   t6Attributes                   [2]IMPLICIT T6Attributes OPTIONAL }

PortionReferenceName              ::= [APPLICATION 0]IMPLICIT PrintableString

TypeOfCoding                      ::= INTEGER {t6 (1)}

T6Attributes                      ::= SET {
   numberOfPelsPerLine            [0]IMPLICIT INTEGER OPTIONAL,
   compression                    [2]IMPLICIT Compression OPTIONAL,
   numberOfDiscardedPels          [3]IMPLICIT INTEGER OPTIONAL}

TextInformation                   ::= CHOICE{
   t6String                       OCTET STRING }
```

Table A.10: *Formal definition of text units.*

Appendix B

Document Architecture Transfer And Manipulation

In this appendix DATAM will be discussed. DATAM is the integrated approach for the data transfer of telematic services. This means that every telematic service is not specified separately, but is seen as a special form (application profile) of DATAM. This approach of telematic services is intended for use over the Integrated Services Digital Network (ISDN). Furthermore, the structural approach of DATAM facilitates interworking between different services. On the other hand ISDN is not being developed for telephony service only, but for more extended services (like FAX and videotex). Therefore the development of DATAM contributes to the development of the ISDN.

B.1 Document Transfer and Manipulation

In this section the communication part of DATAM, i.e. DTAM, is described. In the next sections ODA, which forms the document part of DATAM, is described.

B.1.1 General characteristics of DTAM

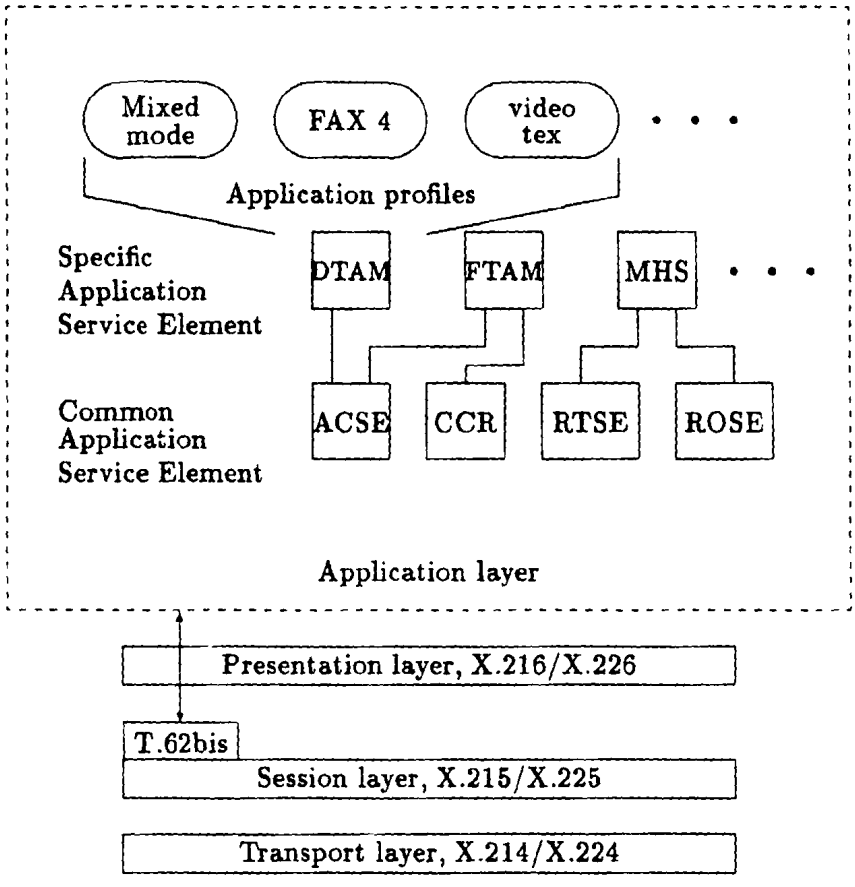
The T.430 series of recommendations define a DTAM service and specify a DTAM protocol available within the application layer of the OSI model. DTAM is a Specific Application Service Element (SASE) of this application layer. Other SASEs are for instance File Transfer Access and Management (FTAM) and Message Handling Systems (MHS). In Figure B.1 a simplified example of an application layer is given.

DTAM puts the document in a communication environment, where the following services can be offered:

- document transfer;
- document manipulation;
- document access and
- document management.

For these services the following communication functions are specified:

- **Document bulk transfer,** which is subdivided into two functionalities:
  - direct document bulk transfer
  - indirect document bulk transfer (uses MHS X.400 series);
- **Remote document manipulation.** Five operations for manipulation can be distinguished :



ACSE = Association Control Service Element
CCR = Commitment Concurrency Recovery
RTSE = Reliable Transfer Service Element
ROSE = Remote Operation Service Element

Figure B.1: *The place of DTAM in the OSI model.*

- create operation,
- delete operation,
- modify operation,
- call operation and
- rebuild operation (for further study by CCITT);

- **Remote document access** (for further study by CCITT);
- **Remote document management** (for further study by CCITT) and
- **Communication support functions.** DTAM uses the following services to exchange protocol elements between DTAM protocol machines:
  - the service of the session layer or

– the service of ACSE[1] and the service of the presentation layer.

To obtain compatibility with the previous CCITT Recommendations in the field of document interchange (T.73), two modes are introduced:

- Transparent mode: in which case there is no use of a presentation layer (like T.73) and
- Normal mode: in which case a presentation layer is used.

B.1.2 Communication application profile

Since there are a number of possible applications (e.g. FAX 4, videotex, etc), for every application a profile is defined. This is called the DATAM application profile (see Figure B.1). The DATAM application profile consists of a communication application profile and a document application profile[2]. The profiles for FAX 4 are discussed in section 4.2.

Here the communication application profile is discussed in general. In the next section the document application profile will be described.

A communication application profile must specify:

- a service class;
- functional units and
- communication support functions.

Service classes

A service class indicates which document interchange facilities of DTAM a specific application utilizes. The following three service classes are defined:

- Document bulk transfer (direct);
- Document manipulation and
- Document bulk transfer and manipulation.

[1]ACSE = Association Control Service Element. Other common application service elements like RTSE or ROSE (see Figure B.1) are for further study by CCITT

[2]Optionally an operational structure profile can be specified.

There may be other service classes such as document bulk transfer (indirect), remote document access and remote document management. These service classes are left for further study (by CCITT).

Functional units

DTAM services are grouped into functional units. During the set-up of the association, the functional units which will be used are negotiated (except the kernel which is always used). The following functional units can be distinguished:

Association use control functional unit (kernel): This unit supports basic DTAM services for selection of functional units, establishment, termination and abort of association use.

Capability functional unit: This unit provides the means for invocation or negotiation of application and communication characteristics.

Data transmission functional unit: This unit consists of the following three units:

1. Document bulk transfer functional unit, the document represented by ODIF is transmitted using this unit.
2. Document manipulation functional unit (confirmed or unconfirmed), to manipulate structures in an existing document.
3. Typed data transmission functional unit, transmits commands for document filing, retrieval and interrupt without subjecting to token control.

Session management functional unit: This unit consists of two functional units, which are control functions for conversational control provided by the session layer. These units are:

- Token control functional unit, controls transmission rights for document transfer and document manipulation;
- Reliable transfer support functional unit; DTAM provides the Activity control, synchronization/resynchronization control unit and error recovery control unit.

Exception report functional unit: This unit provides a reporting service for exceptional conditions occurred in DTAM communication environment.

Remote document access functional unit: for further study by CCITT.

Remote document management functional unit: for further study by CCITT.

Other functional units: DTAM may use ROSE (X.219), RTSE (X.218) and MHS (X.400 series). They are for further study (by CCITT).

Communication support functions

These were mentioned in the previous subsection. In Table B.1 valid combinations of service classes and communication support functions are given.

Service classes		DTAM communication support functions
Document bulk transfer (direct)	DB_0	Direct mapping to session service
	DB_1	ACSE and presentation service
	DB_2	ACSE and RTSE and presentation service (note)
Document manipulation	DM_1	ACSE and presentation service
Document bulk transfer and manipulation	DBM_1	ACSE and presentation service (note)
	DBM_2	ACSE and RTSE and presentation service (note)

note: These service classes are left for further study

Table B.1: *Valid combinations of service classes and communication support functions.*

B.2 Open Document Architecture

In the introduction the Open Document Architecture, ODA, contained by DATAM was mentioned. ODA was developed to facilitate the interchange of documents, between dissimilar systems in an OSI environment. By introducing a common model for these documents, every system within the network only needs two convertors (from the internal document architecture to ODA and vice versa).

In this section the general principles of ODA are discussed and as mentioned in the previous section, the document application profile is described. We will also take a look at the document processing model.

B.2.1 General principles of ODA

ODA is specified by the T.410 CCITT Recommendations, and specifies a document architecture that provides for the representation of documents in three forms:

- formatted form, that allows documents to be presented as intended by the originator;
- processable form, that allows documents to be edited and formatted as intended by the originator or

- formatted processable form, that allows documents to be presented as well as edited and reformatted in accordance with the originators intentions.

The document architecture consists of rules for defining:

- the structure of a document and
- the representation of documents.

The structural model and the descriptive representation form complementary views of a document. The structural model describes the structural elements of a document and their relationships. The descriptive representation describes how the elements of a document and the properties of these elements are represented. These elements are called constituents and they consist of a set of attributes (described in section B.2.4). Those constituents that form counterparts of elements of the structural model are called descriptions. Additional information, such as styles and the document profile, is also represented by the descriptive representation. The constituents form the actual information that is interchanged. The structural model on the other hand is included to:

- delimit portions within a document;
- delimit portions of a document that have a logical meaning;
- use different coding types for different content types and
- allow processing of the document.

In Figure B.2 the relationship between the structural model and descriptive representation is shown. The various elements and constituents mentioned in this figure are subject of discussion in the following subsections.

B.2.2 The structural model of a document

The ODA model divides the document into two parts: the document profile and the document body. The document profile describes the characteristics which are valid for the document as a whole (such as non-basic document characteristics and architecture features). The document profile is not part of the structural model. In the next subsections the document body, as described by the structural model, is discussed.

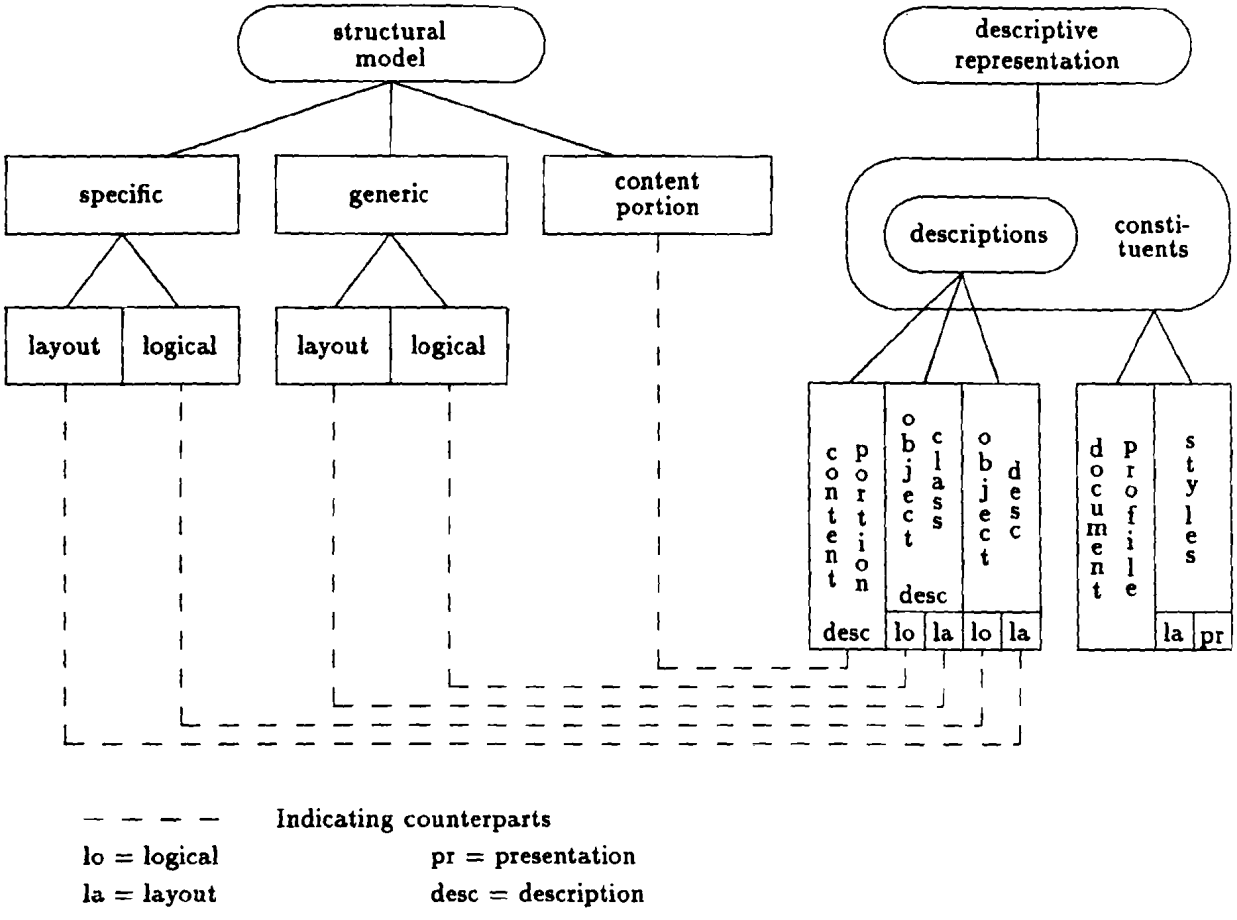


Figure B.2: *Relationship between structural model and descriptive representation.*

Logical and layout structure

Two structures, that provide alternative but complementary views of the same document, can be applied to a document. These structures are:

- Layout structure; division of a document on basis of layout and
- Logical structure; division of a document on basis of meaning.

Both structures consist of objects (= structural elements). Distinction can be made between basic objects, which can not be subdivided into smaller objects, and composite objects, which are subdivided into smaller objects.

The layout structure consists of the following layout objects:

- block (basic)

- frame (composite)
- page (basic or composite)
- page set (composite)

The layout structure is given in Figure B.3.

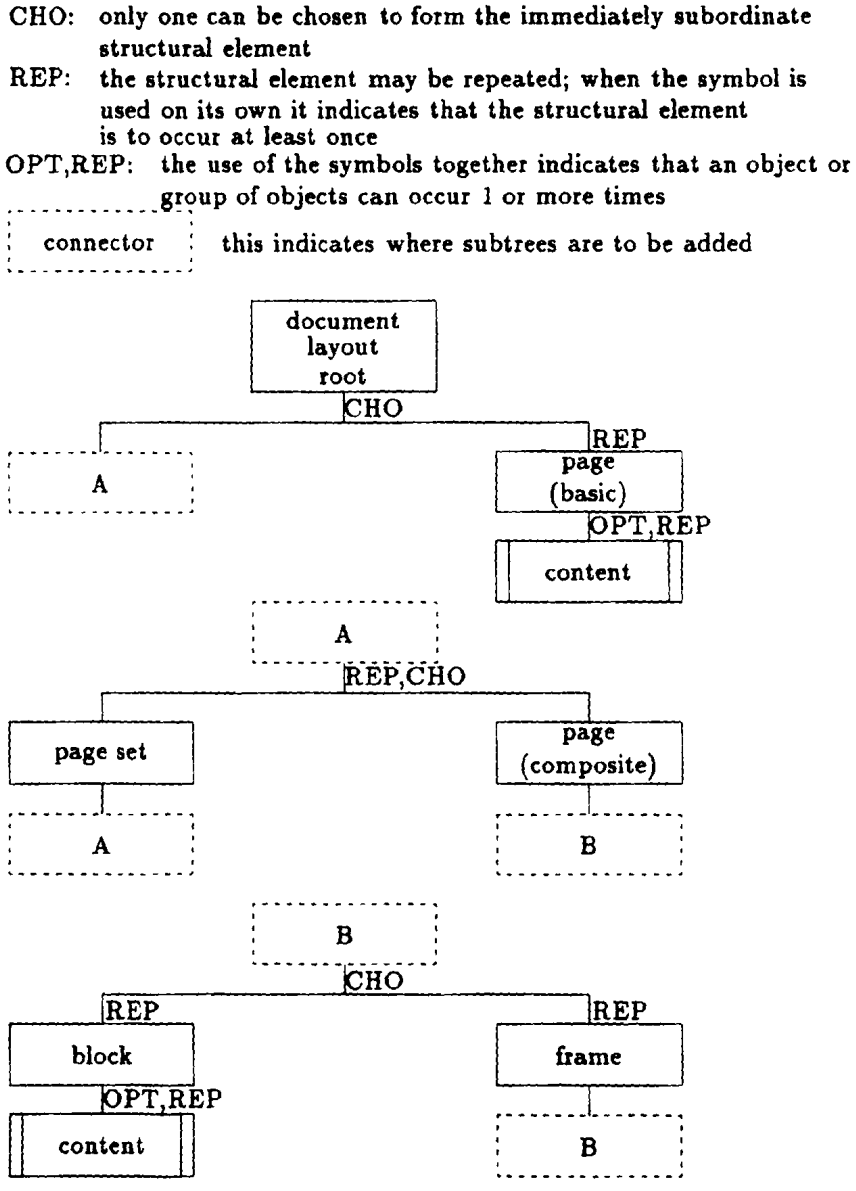


Figure B.3: *Permissible layout structures.*

There is no classification for logical objects, only basic or composite logical objects are distinguished. For further details reference is made to the relevant recommendation (T.412).

Content portions and content architectures

A basic logical/layout object is associated with one or more content portions. A content portion consists of a set of related content elements (basic element of the content), and can only belong to one basic logical object (if any) and to one basic layout object (if any).

The content of the basic logical/layout object is structured according to only one content architecture. This means that the content of a content portion is also structured according to only one content architecture.

At the moment three content architectures are specified:

- Character content architecture (Recommendation T.416)
- Raster graphics content architecture (Recommendation T.417)
- Geometric graphics content architecture (Recommendation T.418)

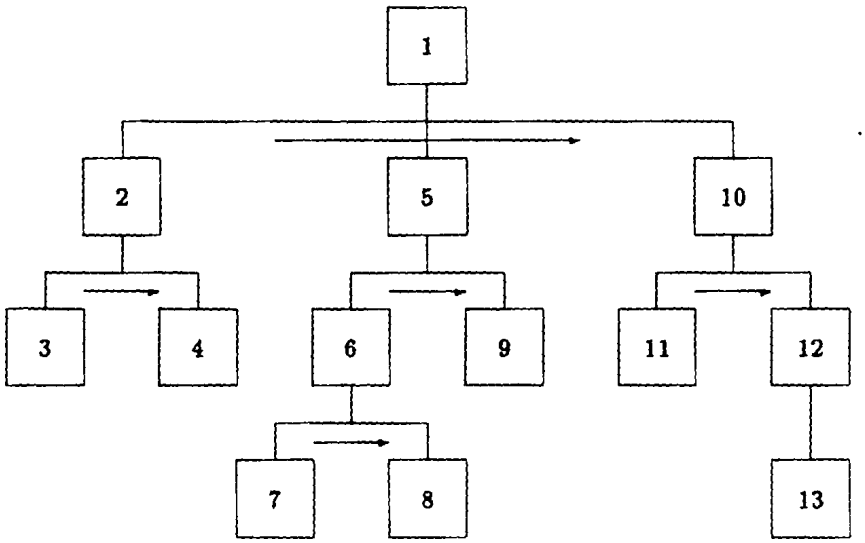
Specific and generic structure

The structures that are particular to a given document are called specific logical structure and specific layout structure. The structural elements of a specific document are called objects. As mentioned previously distinction can be made between basic and composite objects. In the case that more than one immediate subordinate is identified by a composite object, then an ordering of these immediate subordinates is specified. This is called the sequential order. The sequential order of the specific logical/layout structure is called the sequential logical/layout order. In Figure B.4 an example of a sequential order is given.

A group of similar objects (documents) is called an object class (document class). The set of logical object classes and layout object classes, associated with a document, and their relations are called generic logical structure and generic layout structure.

The generic structure can be used to:

- improve transmission efficiency;
- maintain the internal consistency of a document by providing the recipient with the structural information necessary to edit and/or layout the document as intended by the originator.
- facilitate the creation of objects and documents by the recipient as prepared by the originator.



Note: arrows define the sequential order

Figure B.4: *Example of a sequential order.*

Content portions associated with an object class are called generic content portions.

B.2.3 The descriptive representation of a document

For the purpose of interchange, a document is represented as a collection of constituents, each of which is a set of attributes. Those constituents that are counterparts to the structural elements are called descriptions.

Below, the constituents of a document are described:

- document profile, set of attributes specifying the characteristics of a document as a whole;
- logical object description, set of attributes specifying an object within a structure;
- layout object description, as logical object description;
- logical object class description, set of attributes specifying an object class within a document;
- layout object class description, as logical object class description;
- presentation style, set of attributes which guide the format and appearance of the content of the document on the presentation medium;

- layout style, set of attributes which guide the creation of a specific layout structure during document layout and
- content portion description, set of attributes specifying a content portion within the document.

B.2.4 Attributes

An attribute is a property of a document, or of a document component[3], that expresses a characteristic of the document or document component concerned, or a relationship between one or more documents or document components. Each attribute is identified by a name and has a value that describes this characteristic or relationship.

The attributes can be categorized as follows:

- document profile attributes;
- component description attributes, which can be subdivided into:
  - shared attributes;
  - logical attributes;
  - layout attributes;
- layout style attributes;
- presentation style attributes and
- content portion attributes.

The presentation attributes, which can be specified in presentation styles or in basic layout component descriptions, depend on the content architecture[4]. pertaining to a component description. All the other attributes are independent of the content architecture.

The attribute values can only consist of data elements. However the attributes of component descriptions and layout styles may also have an expression as their value. Furthermore, attributes can be classified as mandatory, non-mandatory or defaultable. The latter may be specified in a default value list, which can be specified for composite component descriptions only, and applies to *all* subordinate objects (e.g. a default value list for a page applies to frames and blocks within that page).

To determine the value of an attribute of *objects* classified as defaultable, use the first rule which is applicable:

[3]The term component is used to indicate an object or an object class

[4]These sets are specified in T.416, T.417 and T.418

1. Value specified in the object description concerned;
2. Value specified in a style to which the object description concerned refers;
3. Value specified in an object class description to which the object description concerned refers;
4. Value specified in a style to which an object class description, referred to by the object description concerned, refers;
5. Value specified in an object class description of a resource document[5] to which an object class description, referred to by the object description concerned, refers;
6. Value specified by a style to which an object class description of a resource document is referring. The object description concerned is referring to an object class description, which is referring to that object class description of the resource document;
7. Value specified in a default value list at a higher level of the hierarchical structure. If more than one default value list specifies a value for the same attribute, then the value derived from the lowest hierarchical level in the structure is used. The default value list of an object description has higher priority than the default value list of an object class description, which again has higher priority than a default value list of an object class description of a resource document;
8. Value specified by the document profile attribute "document application profile defaults";
9. Value specified in the CCITT Recommendations.

To determine the value of an attribute of a *content portion* classified as defaultable, use the first rule which is applicable:

1. Value specified by the content portion description;
2. Value specified in the document profile by the attribute "document application profile defaults";
3. Value specified in the CCITT Recommendations.

As mentioned, some attributes may have an expression as their value. There are four types of expressions:

- String expressions;

[5]A resource document is a generic document. When an object class description in a given interchange document contains a reference to an object class description external to that document, then the referred document is called the resource document.

- Numeral expressions;
- Object identifier expressions and
- Construction expressions.

Since the description of these expressions is rather extensive and not of interest for the rest of this study, only a reference to the appropriate recommendation, [11, section 5.1.3 of T.412], is made.

Note: In principle the logical and layout structures are independent of each other. The logical structure is determined by the author, the layout structure is determined by the formatting process. This formatting process is controlled by the attributes called layout directives. These attributes may be associated with the logical structure (e.g. a new chapter starts on a new page).

B.2.5 Document architecture classes

A document architecture class is a set of rules defining the structure and representation of documents in formatted form, processable form or formatted processable form. This means three classes can be distinguished:

Formatted Document Architecture class, (FDA): This class includes:

- document profile;
- specific layout structure;

Optionally it can include:

- presentation styles;
- generic layout structure.

Processable Document Architecture class (PDA): This class includes:

- document profile;
- specific logical structure;

Optionally it can include:

- generic logical structure;
- generic layout structure;
- layout styles;
- presentation styles.

Formatted Processable Document Architecture class (FPDA): This class includes:

- document profile;
- specific logical structure;
- specific layout structure;
- generic layout structure;

Optionally it can include:

- generic logical structure;
- layout styles;
- presentation styles.

B.2.6 Document processing model

The document processing model shows three processes:

– **editing process;**
– **layout process;**
– **imaging process.**

The editing process includes creation and revision of a document. The process edits the content and the logical structure.

The layout process includes the document layout process and the content layout process. The process is concerned with the creation of a specific layout structure.

The imaging process takes a specific layout structure, a corresponding generic layout structure with associated formatted content portions and information contained in presentation styles, and displays it on a suitable presentation medium.

B.2.7 Document application profile

A document application profile is a specification of features to select that part of ODA which is necessary for the support of a particular application. These features are:

- features pertaining to a document architecture class are selected to form a *document architecture level*;
- features of a content architecture class are selected to form a *content architecture level*;

- features of the document profile are selected to constitute a *document profile level* and
- an *interchange format class* is selected.

The document architecture features can be broken down into three classes (FDA, PDA or FPDA), for each class its constituents, for each constituent its attributes and for every attribute its classification, permissible values and default values.

The content architecture features depend on the type of content. For each type of content various content architecture classes exist. For each content architecture class its presentation attributes, coding attributes and control functions, and for each of them their permissible values and default value are defined.

The features of the document profile are its attributes and for each attribute its classification and permissible values.

The interchange formats that are permitted are ODIF class A and class B.

Appendix C

Establishment procedure between a 1984 and 1988 implementation

This appendix describes the establishment of an association between a facsimile group 4 implementation according to the 1984 CCITT Recommendations and a facsimile group 4 implementation according to the 1988 CCITT Recommendations. Two different situations can be distinguished:

1. The 1988 implementation is the sender and the 1984 implementation is the receiver (see Figure C.1).
2. The 1984 implementation is the sender and the 1988 implementation is the receiver (see Figure C.2).

Note: A dashed vector in these figures indicates that the parameters are not mapped onto each other, but that one parameter is derived from the other.

First situation:

The FAX 4 application initiates a D-INITIATE request DTAM service primitive. The primitive consists of three fields:

- Transparent Mode (TM). This field is only used locally, i.e. it indicates to the DTAM Protocol Machine (DTAM-PM) that the transparent mode (no presentation layer) is used.
- Telematic Requirements (TR). This field is used to indicate the functional units available at the sender.
- Application Capabilities. This field indicates the document application profile and document architecture class used by the user (i.e. FAX 4).

This primitives forms, together with information stored in the DTAM-Protocol Machine, a DINQ APDU. Only the application Capabilities of the primitive are mapped onto the APDU.

The DTAM-PM initiates a S-CONNECT request session service primitive. This primitives consists of:

- Interface Control Information (ICI). This parameter is used to control the information transfer at a Service Access Point (SAP).
- Session Requirements (SR). This parameter indicates the functional units selected by the DTAM-PM, i.e. the DTAM-PM uses the Telematic Requirements of the D-INITIATE req primitive to select the Session Requirements.
- User Data. This parameter is used to convey data of the application layer, i.e. the DINQ APDU.
- Other parameters (not of interest for this description).

The S-CONNECT primitive asks the session layer to provide for the establishment of a connection. To achieve this service, the session protocol machine forms a Command Session Start, i.e. "PDU". This CSS consists of:

- A header (Head).
- The parameter Session Service Functions (SSF). This parameter is used to interchange the functional units used by the sender.
- Session User Data. This parameter is used to convey data of the application layer, i.e. the APDU.
- Other parameters (not of interest for this description).

The CSS is transferred to the receiver using the lower layers of the OSI Reference Model.

When arriving at the receiver the CSS command initiates a S-CONNECT indication session service primitive. This primitive is equal to the S-CONNECT request primitive.

Since the receiver is a 1984 implementation it uses no APDUs and D-primitives. The negotiation (of e.g. the functional units, storage capacity, etc) is completely done by the session layer. The application (in 1984 this is called presentation) capabilities which are to be negotiated, are obtained from the application layer through protocol elements.

The S-CONNECT response session service primitive is initiated by the Document Interchange Protocol (DIP). It consists of:

- ICI.
- The parameter Result (Rslt). When the connection establishment request fails, this result will be negative. Otherwise the result will be positive. In the former case the session protocol machine initiates a Response Session Start Positive (RSSP), in the latter case it initiates a Response Session Start Negative (RSSN).
- The parameter SR indicates the functional units available at the receiver.
- The parameter User Data conveys the application capabilities of the receiver.
- Other parameters.

In case of a negative result the reason for the rejection is indicated in the parameter Reason (Rsn). Otherwise RSSP is initiated. This PDU has no result parameter, because receipt of the PDU indicates that the result is positive.

The RSSP/RSSN initiates a S-CONNECT confirm session service primitive. This primitive has a result parameter, which depending on the SPDU is positive or negative.

The DTAM-PM forms a D-INITIATE confirm primitive using the "Result" and "Session Requirements" parameters of the S-CONNECT confirm primitive and the DINR APDU.

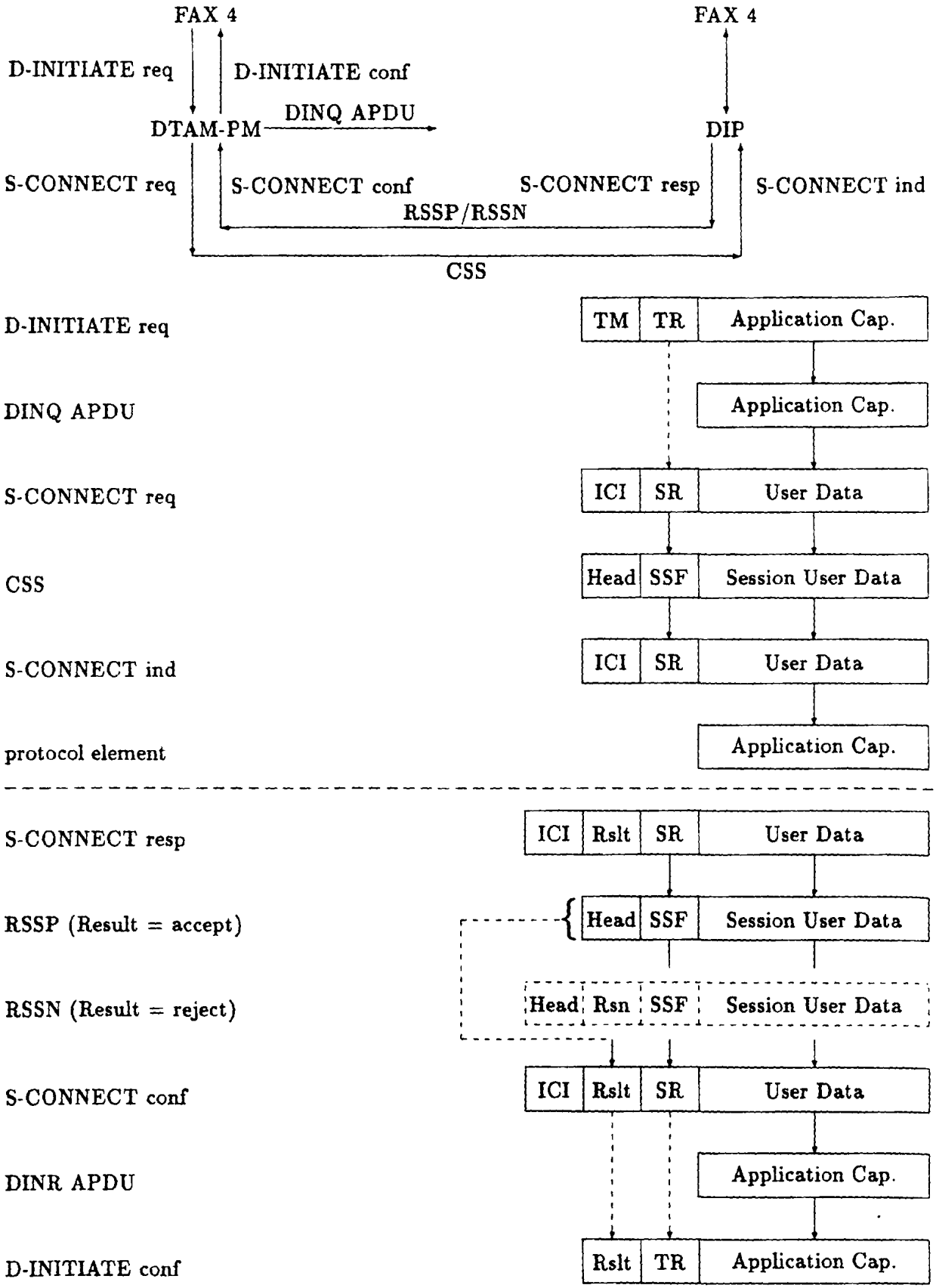


Figure C.1: *Establishment procedure (1988 implementation is sender).*

Second situation:

The DIP initiates a S-CONNECT request. The User Data of this primitive contains the application capabilities of the user.

The primitive initiates the CSS PDU.

At the receiver this PDU initiates a S-CONNECT indication primitive. The user data is mapped onto the DINQ APDU.

The DTAM-PM initiates the D-INITIATE indication primitive.

The receiver initiates the D-INITIATE response primitive. This consists of:

- Result parameter, indicating acceptance or rejection of the capabilities.
- Telematic requirements parameter.
- Application Capabilities parameter.

This primitive forms with data stored in the DTAM-PM the DINR APDU, consisting of the field Application Capabilities.

This PDU is interchanged by the session service primitive S-CONNECT response. The result parameter of this S-CONNECT response primitive indicates the acceptance or rejection of the session connection. Rejection will occur when the receiving DTAM-PM detects an error in the S-CONNECT indication primitive or when the receiver rejects the application-association (indicated by the Result parameter value 'rejected' of D-INITIATE response).

Depending on this parameter a RSSP or RSSN PDU is transferred.

At the sender this PDU initiates a S-CONNECT confirmation.

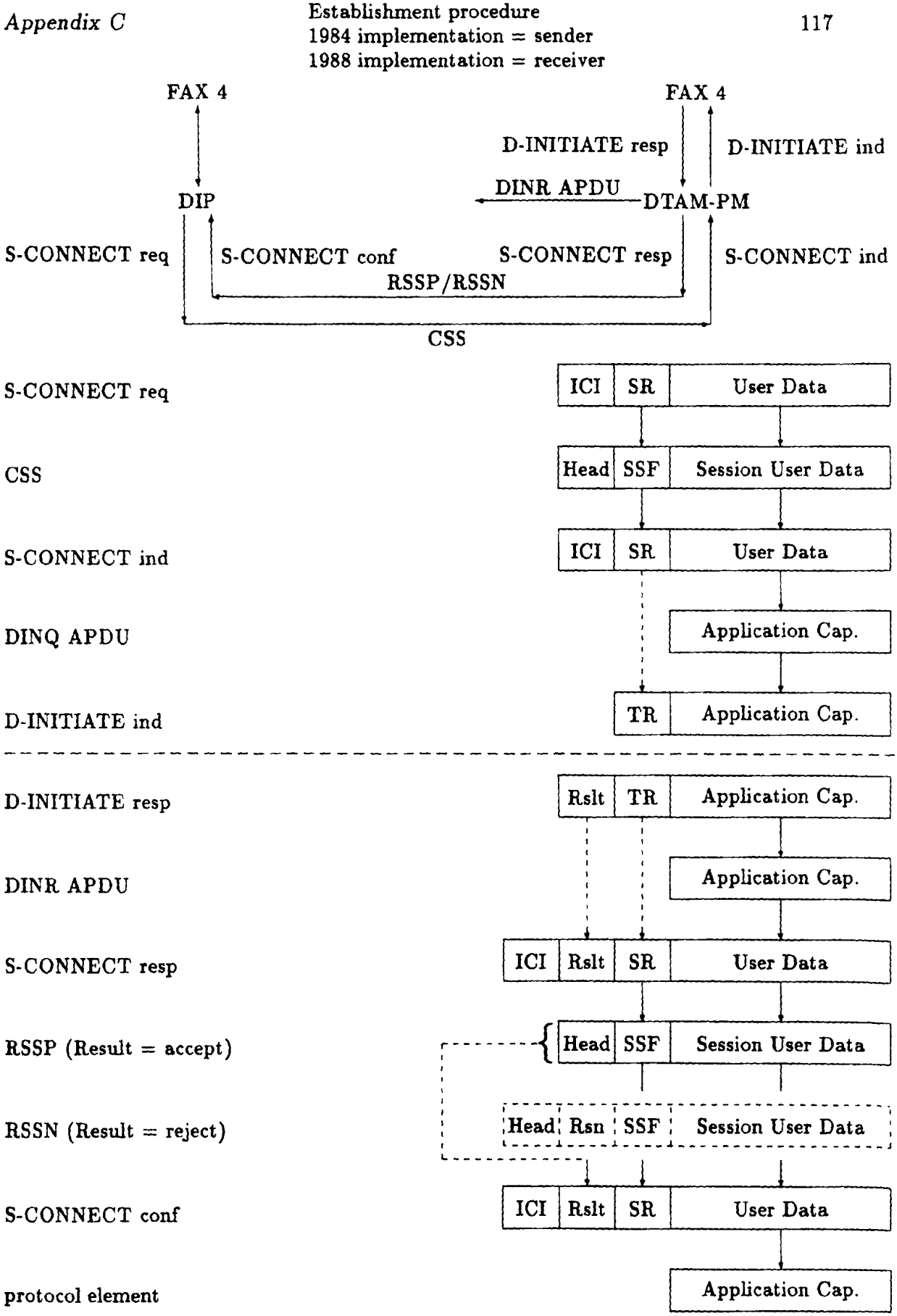


Figure C.2: *Establishment procedure (1984 implementation is sender).*

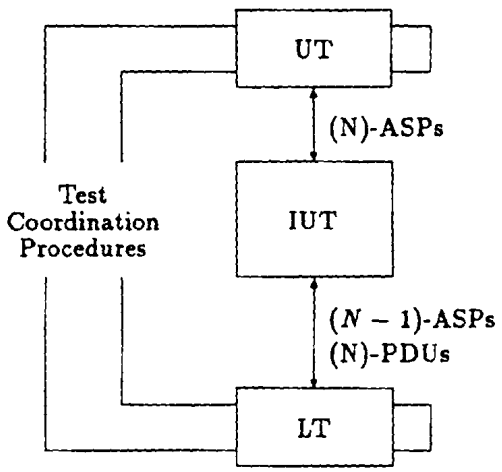
Appendix D

Test methods overview

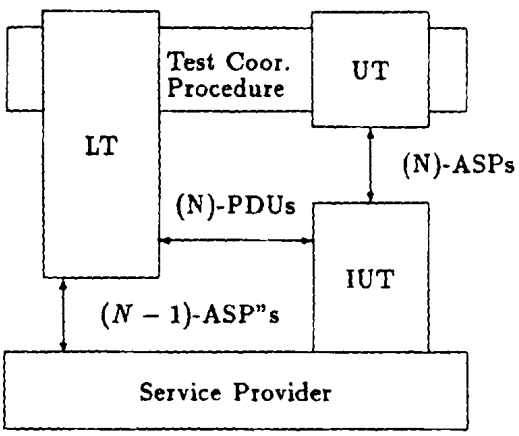
In this appendix the various test methods are illustrated. For end systems only the single-layer test methods are clarified. In case of multi-layer testing, the IUT consists of more than one layer. This means that not only the PDUs of that layer, but also the PDUs of the other layers above have to be observed and controlled.

In the relay-systems N_t means the top layer. This can be either layer 3 or layer 7. N_b means the bottom layer, which is always layer 1.

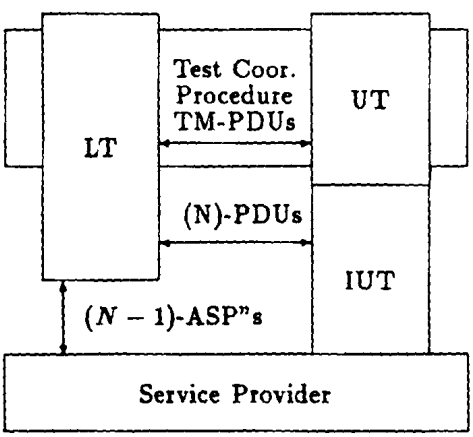
In Table D.1 an overview of the terms, which specify the test events for the various test methods, is given.



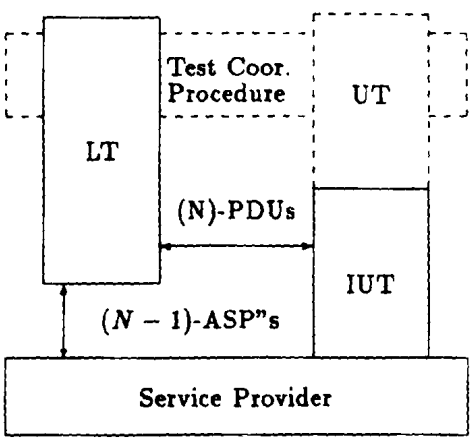
The LS Test Method



The DS Test Method

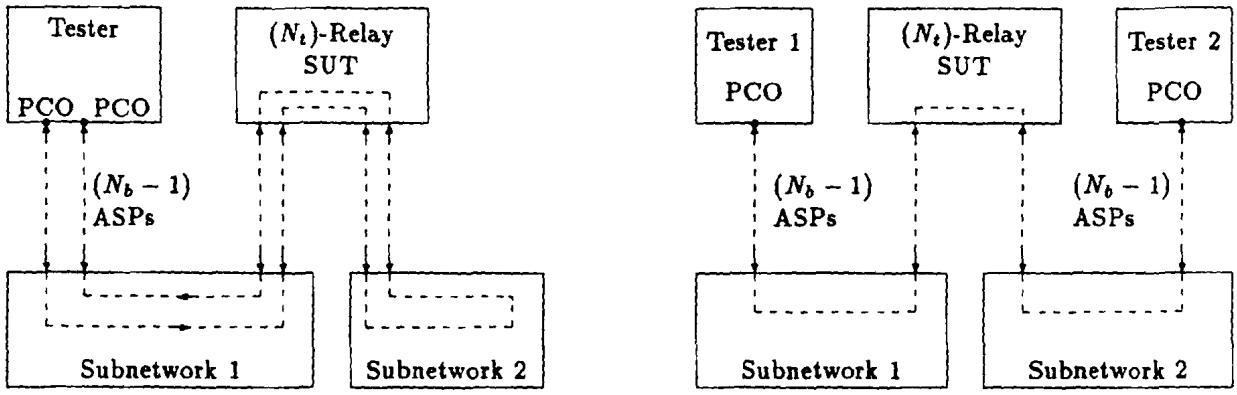


The CS Test Method



The RS Test Method

Figure D.1: *Test method overview.*



The Loop-back Test Method (YL)

The Transverse Test Method (YT)

Figure D.2: *Test method overview (continue).*

DTAM service primitive	APDU
D-INITIATE	DINQ and DINR
D-TERMINATE	DTEQ and DTER
D-ABORT	DAB
D-CAPABILITY	DCPQ and DCPR
D-TOKEN-PLEASE	DTP

Table D.1: *Test methods for end-systems and the terms specifying them.*

Appendix E

Test suite

E.1 Declaration part

```
$Suite Test_FAX4_84_88_Compatibility

$Declarations

$ConstDcls
 LTA:   INTEGER '00'H;
 DRN:   INTEGER 1;
 DI:    INTEGER '0000'H
$EndConstDcls

$VarDcls
 prn, prnc: INTEGER 1;
 compat : BOOLEAN FALSE
$EndVarDcls

$PCODcls
 LT: K1195;
 OUT: Disc
$EndPCODcls

$ASPDcls
 $ASP S_CONreq (CallingTermId, Date_Time, ServId: INTEGER; SUD: Protel)
 $PCOTypes K1195;
 $ASP S_CONconf (CalledTermId, Date_Time, ServId: INTEGER; SUD: Protel)
 $PCOTypes K1195;
 $ASP  S_ASreq (AI: INTEGER; SUD: Protel)
 $PCOTypes K1195;
 $ASP S_MSconf (SPSN: INTEGER; SUD: INTEGER)
 $PCOTypes K1195;
 $ASP S_U_EXPTind (Reason: INTEGER)
 $PCOTypes K1195;
 $ASP S_P_EXPTind (Reason: INTEGER; Reflect: Protel)
 $PCOTypes K1195;
 $ASP S_AI (Reason: INTEGER)
 $PCOTypes K1195;
 $ASP S_ARESreq (OAI,SPSN,AI: INTEGER; SUD: Protel)
 $PCOTypes K1195;
 $ASP S_U_ABT
 $PCOTypes K1195;
 $ASP S_DAreq (SUD: Protel)
 $PCOTypes K1195;
 $ASP S_MSreq (SPSN: INTEGER)
 $PCOTypes K1195;
 $ASP S_COMPAT (Compat: BOOLEAN)
 $PCOTypes Disc
$EndASPDcls
```

```
$PDUTypeDcls
$PDUTypeIds Protel
Protel ::= CHOICE
  { conresprotel          Conresprotel,
    dataprotel            Dataprotel
  }

Conresprotel ::= CHOICE
   { presCapDesc [4]IMPLICIT PresCapDesc
   }

Dataprotel ::= SEQUENCE
  { oneOrTwoLayoutprotel      OneOrTwoLayoutprotel,
    textUnitprotel            [3]IMPLICIT TextUnitprotel
  }

PresCapDesc ::= SET
  { basicTermChar         [0]IMPLICIT BasicTermChar,
    interForm             [1]IMPLICIT InterForm,
    nonBasicTermCap       [2]IMPLICIT NonBasicTermCap OPTIONAL,
    contArchClasses       [5]IMPLICIT INTEGER OPTIONAL, -- SET OF PrintableString
    formClass             [6]IMPLICIT INTEGER OPTIONAL
  }

OneOrTwoLayoutprotel ::= CHOICE
  { specLayoutprotel          [2]IMPLICIT SpecLayoutprotel,
    twoSpecLayoutprotel       TwoSpecLayoutprotel
  }

TwoSpecLayoutprotel ::= SEQUENCE
  { firstSLprotel         [2]IMPLICIT SpecLayoutprotel,
    secondSLprotel        [2]IMPLICIT SpecLayoutprotel
  }

BasicTermChar ::= OCTET STRING

InterForm ::= OCTET STRING

NonBasicTermCap ::= SET
  { pageDim     [2]IMPLICIT SEQUENCE OF MeasurePair OPTIONAL,
    codingAtt   [3]IMPLICIT SEQUENCE OF CodingAtt OPTIONAL,
    presAttsP   [4]IMPLICIT SEQUENCE OF PresAttsP OPTIONAL
  }

MeasurePair ::= SEQUENCE
  { hmes        Mes,
    vmes        Mes
  }
```

```
CodingAtt ::= CHOICE
  { compres       [0]IMPLICIT Compres
  }

PresAttsP ::= CHOICE
  { pelTranDen    [11]IMPLICIT PelTranDen
  }

Mes ::= CHOICE
  { fixed         [0]IMPLICIT INTEGER,
    var           [1]IMPLICIT INTEGER
  }

Compres  ::= INTEGER{uncomp(0),comp(1)}

PelTranDen  ::= INTEGER{p200 (1),p240 (2),p300 (3),p400 (4)}

SpecLayoutprotel ::= SEQUENCE
   { layoutObType        LayoutObType,
     layoutDescBody      LayoutDescBody OPTIONAL
   }

LayoutObType ::= INTEGER{doc (0), page (2)}

LayoutDescBody ::= SET
    { obId               ObRefName OPTIONAL,
      refTo              CHOICE
      { subOr                  [0]IMPLICIT INTEGER, -- SEQUENCE OF NumString
        cont                   [1]IMPLICIT INTEGER -- SEQUENCE OF NumString
      } OPTIONAL,
      dim                [4]IMPLICIT MeasurePair OPTIONAL,
      presAtts           [6]IMPLICIT PresAtts OPTIONAL,
      defaultVL          [7]IMPLICIT SEQUENCE OF DefaultVL OPTIONAL
     }

ObRefName ::= [APPLICATION 1]IMPLICIT INTEGER -- PrintableString

PresAtts ::= SET
    { contType           ContType OPTIONAL,
      photoAtts          [1]IMPLICIT PhotoAtts OPTIONAL
    }

DefaultVL ::=  CHOICE
    { pageAtt            [2]IMPLICIT PageAtt
    }

ContType ::= [APPLICATION 2]IMPLICIT INTEGER
```

```
PhotoAtts ::= SET
  { pelPath       [0]IMPLICIT OneOfFourAngles OPTIONAL,
    linePro       [1]IMPLICIT OneOfTwoAngles OPTIONAL,
    pelTranDen    [2]IMPLICIT PelTranDen OPTIONAL
  }

OneOfFourAngles ::= INTEGER{d0 (0)}

OneOfTwoAngles ::= INTEGER{d270 (3)}

PageAtt ::= SET
  { dim           [1]IMPLICIT MeasurePair,
    presAtts      [3]IMPLICIT PresAtts
  }

Att ::= CHOICE
  { dim           [1]IMPLICIT MeasurePair,
    presAtts      [3]IMPLICIT PresAtts
  }

TextUnitprotel ::= SEQUENCE
     { contPortAtt        ContPortAtt OPTIONAL,
       textInfo           TextInfo
     }

ContPortAtt ::= SET
     { contPortId         PortRefName OPTIONAL,
       typeOfCoding       TypeOfCoding OPTIONAL,
       t6Att              [2]IMPLICIT T6Att OPTIONAL
     }

TextInfo ::= CHOICE
 { t6String OCTET STRING
 }
PortRefName ::= [APPLICATION 0]IMPLICIT INTEGER -- PrintableString

TypeOfCoding ::= INTEGER

T6Att  ::= SET
     { noOfPel_Line       [0]IMPLICIT INTEGER OPTIONAL,
       compres            [2]IMPLICIT Compres OPTIONAL,
       noOfDiscPel        [3]IMPLICIT INTEGER OPTIONAL
     }

$EndPDUTypeDcls
```

$TimerDcls
T1: inactivityTimer 60 sec;
T2: demand_responseTimer 60 sec
$EndTimerDcls

$EndDeclarations

$Comment
Declaration of the variables:

- prn = page reference number and
- prnc = page reference number confirmed.

Declaration of constants:

- LTA = Lower Tester Adress
- DRN = Document Reference Number;
- LTE = Local Terminal Error and
- DI = Document Information (i.e. t6String of actual contents).

Decalaration of Points of Control and Observation:
LT = Lower Tester (i.e. Siemens K1195 protocol analyser)
OUT = predefined memory location (e.g. harddisk of K1195)

Declaration of Abstract Service Primitives:
In this part the various service primitives which can be controlled and observed by the lower tester are introduced. Every primitive is mentioned with its parameters, which are given a value in the constraints part.

Declaration of the Protocol Data Units:
The declaration of the PDUs is the same as the ASN.1 description in T.73 or T.415/T.417. However the option oneOrTwoLayoutprotel is introduced. This option is needed because the user data parameter of the first S-DATA request contains a document descriptor and a page descriptor. The other S-DATA requests only transfer a page descriptor.

Declaration of Timers:

- T1 = inactivity timer of the session implementation, which has 60 sec as default and
- T2 = demand/response timer, i.e. the time between sending and receiving a PDU is 60 sec maximum.

$EndComment

E.2 Constraints part

```
$Constraints
 $ASPConstraints

  $Constraint S_CONNECTreq
   S_CONreq =    {       CallingTermId : LTA,
                         Date_Time : 0,
                         ServId : 1, -- Telematic services (T.62bis sec 4.2.2.2)
                         SUD : data_pe
                 };

  $Constraint S_CONNECTconf(si: INTEGER)
   S_CONconf =   {       CalledTermId : ?,
                         Date_Time : ?,
                         ServId : si, -- Telematic services (T.62bis sec 4.2.2.2)
                         SUD : data_pe
                 };

  $Constraint S_ASTARTreq1
   S_ASreq =     {       AI : DRN,
                         SUD : data_pe
                 };

  $Constraint S_ASTARTreq2
   S_ASreq =     {       AI : DRN,  -- DRN = Document Reference Number
                         SUD : Start_pe1
                 };

  $Constraint S_ASTARTreq3
   S_ASreq =     {       AI : DRN,  -- DRN = Document Reference Number
                         SUD : Start_pe2
                 };

$Comment The protocol analyser is identified by: Lower Tester Address (LTA)
         Service Identifier = 1, i.e. Telematic services (see T.62bis section 4.2.2.2.)
         The S_Connect primitives, must have a parameter Quality Of Service,
         consisting of the subparameters:
         - extended control, i.e. which form of PDU concatenation;
         - optimized dialog transfer.
         Both must have the value "feature not desired" (in case of the request)
         indicating basic concatenation and transport class 0 respectively.
         Both must have the value "agreed" (in case of the confirm), indicating
         acceptance. These values are however not specified in the standards
         (see X.215 Annex B3.1 and T.62bis section 4.2.1.3).
$EndComment
```

```
$Constraint S_ASTARTreq4
 S_ASreq =      {       AI : DRN,  -- DRN = Document Reference Number
                        SUD : Start_pe3
                };

$Constraint S_MINSYNCconf(prnc: INTEGER)
 S_MSconf =     {       SPSN : prnc, -- prnc = page reference number confirmed
                        SUD : 0      -- Further traffic can be accepted
                };

$Constraint S_U_EXPTIONind
 S_U_EXPTind =  {       Reason : 0   -- non specific error (T.62 sec 5.7.2.10)
                };

$Constraint S_P_EXPTIONind
 S_P_EXPTind =  {       Reason : 6,  -- protocol error
                        Reflect : Start_pe1 -- pdu with "ETSI attributes"
                };

$Constraint S_DATAFreq
 S_DAreq =      {       SUD : Dataf_pe
                };

$Constraint S_AINTERreq
 S_AI =         {       Reason : 0   -- non specific error
                };

$Constraint S_AINTERconf
 S_AI =         {       Reason : 0   -- non specific error
                };

$Constraint S_ARESUMEreq(prn: INTEGER)
 S_ARESreq =    {       OAI : DRN,
                        SPSN : prn,
                        AI : DRN,
                        SUD : data_pe
                };
```

$Comment The Session User Data of S_MSconf is 0, meaning that further traffic can be accepted (see T.62 section 5.7.3.6).
The Reason parameter of the primitive S_U_EXCEPTIONind and is 0, meaning that no specific error occurred (see T.62 section 5.7.2.10).
The Reason parameter of S_P_EXCEPTIONind is 6, meaning that a protocol error occurred (see T.62 section 5.7.2.10).
The Reflect parameter of S_P_EXCEPTIONind contains the received PDU up to and including the detected error (see T.62 section 5.7.3.8).
$EndComment

```
$Constraint S_U_ABORTind
 S_U_ABT =     {          -- Local terminal error
               };

$Constraint S_U_ABORTresp
 S_U_ABT =     {          -- Local terminal error
               };

$Constraint S_DATAreq2a
 S_DAreq =     {          SUD : Data_pe2a
               };

$Constraint S_DATAreq2b
 S_DAreq =     {          SUD : Data_pe2b
               };

$Constraint S_DATAreq3
 S_DAreq =     {          SUD : Data_pe3
               };

$Constraint S_DATAreq4
 S_DAreq =     {          SUD : Data_pe4
               };

$Constraint S_MINSYNCreq(prn: INTEGER)
 S_MSreq =     {          SPSN : prn
               };

$Constraint S_COMPATIBLE(compat: BOOLEAN)
 S_COMPAT =    {          Compat : compat
               }

$EndASPConstraints
```

$Comment The reason parameter of S_ACT_INTERRUPTreq/conf is 0, meaning that a non specific error occurred (see T.62 section 5.7.2.10).
The S_MINSYNCreq primitive must include the parameter "Type".
The value of this parameter must be "explicit" in case of FAX 4.
It is not specified how this value should be coded.
S_COMPATIBLE is the only ASP that is send via the PCO "OUT".
$EndComment

$PDUConstraints

```
$Constraint Dataf_pe
  Protel =    { dataprotel :
                    { oneOrTwoLayoutprotel :
                        { specLayoutprotel :
                            { layoutObType : 0  -- value 2 is expected
                            }
                        },
                      textUnitprotel :
                        { textInfo :
                            { t6String : DI
                            }
                        }
                    }
              };
```

$Comment PDU with error (layoutObType should be 2 indicating a page) to initiate a session resume.
$EndComment

```
 $Constraint data_pe
  Protel =    { conresprotel :
                    { presCapDesc :
                        { basicTermChar : '02'H,
                          interForm : '00'H
                        }
                    }
              };
```

$Comment PDU used to establish a session connection.
$EndComment

```
$Constraint Start_pe1
  Protel =  { conresprotel :
                { presCapDesc :
                    { basicTermChar : '02'H,
                      interForm : '00'H,
                      contArchClasses : 2 ,
                      formClass : 1
                    }
                }
            };
```

$Comment PDU used to start a document exchange and coded according to ETSI.
$EndComment

```
$Constraint Start_pe2
  Protel =  { conresprotel :
                { presCapDesc :
                    { basicTermChar : '02'H,
                      interForm : '00'H,
                      nonBasicTermCap :
                        { presAttsP :
                            { { pelTranDen : 2 -- 240 pels/inch
                            } }
                        }
                    }
                }
            };
```

$Comment PDU used to establish a session connection and indicating thatthe sender of this PDU can handle a resolution of 240 pels/inch. This PDU is used to test the compatibility of the coding of the attribute "pel transmission density".
$EndComment

```
$Constraint Start_pe3
  Protel =  { conresprotel :
                { presCapDesc :
                    { basicTermChar : '02'H,
                      interForm : '00'H,
                      nonBasicTermCap :
                        { codingAtt :
                            { { compres : 0 -- uncompressed
                            } }
                        }
                    }
                }
            };

$Comment PDU used to establish a session connection and indicating that the
         sender of this PDU can handle uncompressed coding. This PDU is used to test
         the compatibility of the coding of the attribute "compression".
                                }
```

```
$Constraint Data_pe2b
 Protel =    { dataprotel :
                  { oneOrTwoLayoutprotel :
                      { specLayoutprotel :
                          { layoutObType : 2
                          }
                      },
                    textUnitprotel :
                      { textInfo :
                          { t6String : DI
                          }
                      }
                  }
             };
```

$Comment PDU used to send the second page of information. This PDU will be conveyed in the second S_DATA req and therefor consists of a page descriptor only.
$EndComment

```
$Constraint Data_pe3
 Protel =    { dataprotel :
                  { oneOrTwoLayoutprotel :
                      { twoSpecLayoutprotel :
                          { firstSLprotel :
                              { layoutObType : 0,
                                layoutDescBody :
                                  { refTo :
                                      { subOr : 1
                                      }
                                  }
                              },
                           secondSLprotel :
                              { layoutObType : 2
                              }
                          }
                      },
                    textUnitprotel :
                      { textInfo :
                          { t6String : DI
                          }
                      }
                  }
             };
```

$Comment PDU used to send the first page of information. This PDU will be conveyed in the second S_DATA req and therefor consists of both a document and page descriptor. The PDU is used to test the compatibility of the attribute "reference to subordinates".
$EndComment

```
$Constraint Data_pe4
 Protel =   { dataprotel :
                { oneOrTwoLayoutprotel :
                    { twoSpecLayoutprotel :
                        { firstSLprotel :
                            { layoutObType : 0
                            },
                          secondSLprotel :
                            { layoutObType : 2
                            }
                        }
                    },
                  textUnitprotel :
                    { contPortAtt :
                        { t6Att :
                            { noOfDiscPel : 40  -- 37 expected
                            }
                        },
                      textInfo :
                        { t6String : DI
                        }
                    }
                }
            }
```

$Comment PDU used to send the first page of information. This PDU will be conveyed in the first S_DATA req and therefor consists of both a document and page descriptor. The PDU is used to test the compatibility of the attribute "number of discarded pels".
$EndComment

$EndPDUConstraints
$EndConstraints

$Comment basic terminal characteristics is 2, meaning facsimile group 4;
interchange format is 0, meaning TIF0 or ODIF class B;
content architecture classes is 2 8 2 7 0, meaning formatted raster-graphic content architecture (see T.417 section 6.4.1)
pel transmission density is 2, meaning 240 pels/inch;
compression is 0, meaning uncompressed;
layout object type is 0, meaning document descriptor;
layout object type is 2, meaning page descriptor;
$EndComment

E.3 PDU coding part

```
$PDUCodingDcls
$PDUCodingSet
$DefaultDirection LEFTDEC
 Protel AS CHOICE
        {       conresprotel AS ASN1,
                dataprotel AS Dataprotel
        };

 Dataprotel AS SEQUENCE
        {       oneOrTwoLayoutprotel AS OneOrTwoLayoutprotel,
                textUnitprotel  AS ASN1
        };

 OneOrTwoLayoutprotel AS CHOICE
        {       specLayoutprotel  AS ASN1,
                twoSpecLayoutprotel AS TwoSpecLayoutprotel
        };

 TwoSpecLayoutprotel AS SEQUENCE
        {       firstSLprotel AS ASN1,
                secondSLprotel AS ASN1
        }

$EndPDUCodingSet
$EndPDUCodingDcls

$Comment The coding used is the ASN.1 coding, see [27]
$EndComment
```

E.4 Dynamic part

$DynamicPart

$TestCase Test_FAX4_84_88_Compatibility/Session_Abort/Case_1
$Purpose Test the use of the Session abort service by the 1988 standard
$TestCaseId Sabort

```
<1> LT!S_CONNECTreq
  <2> LT?S_CONNECTconf
    <3> LT!S_DATAreq2a              -- wrong service primitive
      <4> LT?S_U_ABORTind
        <5> LT!S_U_ABORTresp   -- 84 sends abort response
          <6> LT?S_U_ABORTind {compat := FALSE}        $Verdict P
            <7> OUT!S_COMPATIBLE(compat)               $Verdict PASS
          <6> LT?OTHERWISE                             $Verdict FAIL
      <4> LT?OTHERWISE                                 $Verdict FAIL
  <2> LT?OTHERWISE                                     $Verdict FAIL
```

$EndTestCase;

$Comment The purpose of this test case is to test whether a 1988 implementation uses a confirmed or uncomfirmed abort procedure. In the latter case problems of incompatibility may occur, because the 1988 implementation does not expect to receive something back after sending the abort request (except i.e. a connect request). So, a form of deadlock may occur; the 1984 sends a response, the 1988 sees this as an invalid state and sends a request.
$EndComment

```
$TestCase   Test_FAX4_84_88_Compatibility/Session_Resume/Case_2
$Purpose    Test the use of the session resume procedure
$TestCaseId Sresume
 <1> +Preamble [LT]
  <2> {prn := prn - 1}
   <3> LT!S_ARESUMEreq(prn) START(T1)
    <4> +T_433 [LT,OUT]                                    $Verdict PASS
    <4> LT!S_DATAreq2b START(T1)
     <5> +T_433 [LT,OUT]                                   $Verdict PASS
     <5> LT!S_MINSYNCreq(prn) START(T2)
      <6> LT?S_MINSYNCconf(prnc) [prnc = prn] CANCEL(T1;T2)
       <7> {compat := TRUE}                                $Verdict P
        <8> OUT!S_COMPATIBLE(compat)                       $Verdict PASS
      <6> ?TIMEOUT(T2)                                     $Verdict FAIL
      <6> +T_433 [LT,OUT]                                  $Verdict PASS;

$Tree T_433 [LT: K1195; OUT: Disc]
 <1> LT?S_U_ABORTind {compat := FALSE}    $Verdict P
  <2> OUT!S_COMPATIBLE(compat)            $Verdict PASS
 <1> ?TIMEOUT(T1)                         $Verdict FAIL
 <1> LT?OTHERWISE                         $Verdict FAIL

$EndTestCase;


$Comment  The purpose of this testcase is to test whether a 1988 implementation
          has the rigth response to a resume procedure initiated by the 1984
          implementation. At the moment FAX 4 implementations (i.e. according
          to the 1984 recommendations) do not include a resume procedure
          although it should be implemented according to T.62
$EndComment
```

```
$TestCase Test_FAX4_84_88_Compatibility/Ref_to_Subordinates/Case_3
$Purpose  Test the use of the attribute reference to subordinates.
$TestCaseId Subordinates
 <1> LT!S_CONNECTreq
   <2> LT?S_CONNECTconf
     <3> LT!S_ASTARTreq1
       <4> LT!S_DATAreq3     START(T1)
         <5> +result [LT,OUT]                                    $Verdict PASS
         <5> LT!S_MINSYNCreq(prn) START(T1;T2)
           <6> LT?S_MINSYNCconf(prnc) [prnc = prn] CANCEL(T1;T2)
             <7> {compat := TRUE}                                $Verdict P
               <8> OUT!S_COMPATIBLE(compat)                      $Verdict PASS
           <6> +result [LT,OUT]                                  $Verdict PASS
   <2> LT?OTHERWISE                                              $Verdict FAIL;

$Tree result [LT: K1195; OUT: Disc]
 <1> LT?S_U_ABORTind {compat := FALSE}        $Verdict P
   <2> OUT!S_COMPATIBLE(compat)               $Verdict PASS
 <1> ?TIMEOUT(T1)                             $Verdict FAIL
 <1> LT?OTHERWISE                             $Verdict FAIL

$EndTestCase;

$Comment  1984 implementation is tester, 1988 implementation is IUT.
          The purpose of this testcase is to test the use of the attribute
          "reference to subordinates". This attribute can be used by the 1984
          implementation according to T.73. The 1988 implementation on the
          other hand is not able to send or receive this attribute. This means
          that whenever a 1988 implementation receives a PDU containing this
          attribute, it will react to it as to a syntactically invalid PDU.
$EndComment
```

```
$TestCase Test_FAX4_84_88_Compatibility/No_of_Discarded_Pels/Case_4
$Purpose  Test the use of the attribute no of discarded pels.
$TestCaseId Discarded
 <1> LT!S_CONNECTreq
  <2> LT?S_CONNECTconf
   <3> LT!S_ASTARTreq1
    <4> LT!S_DATAreq4   START(T1)
     <5> +result [LT,OUT]                                    $Verdict PASS
     <5> LT!S_MINSYNCreq(prn) START(T1;T2)
      <6> LT?S_MINSYNCconf(prnc) [prnc = prn] CANCEL(T1;T2)
       <7> {compat := TRUE}                                  $Verdict P
        <8> OUT!S_COMPATIBLE(compat)                         $Verdict PASS
      <6> +result [LT,OUT]                                   $Verdict PASS
  <2> LT?OTHERWISE                                           $Verdict FAIL;

$Tree result [LT: K1195; OUT: Disc]
 <1> LT?S_U_ABORTind {compat := FALSE}    $Verdict P
  <2> OUT!S_COMPATIBLE(compat)            $Verdict PASS
 <1> ?TIMEOUT(T1)                         $Verdict FAIL
 <1> LT?OTHERWISE                         $Verdict FAIL

$EndTestCase;

$Comment  1984 implementation is tester, 1988 implementation is IUT.
          See the Comment belonging to testcase Ref_to_Subordinates/Case_3.
$EndComment
```

```
$TestCase Test_FAX4_84_88_Compatibility/ETSIspecification/Case_5
$Purpose  Test the use of the attributes content architecture class
          and interchange format class of the document profile.
$TestCaseId ETSI
 <1> LT!S_CONNECTreq
  <2> LT?S_CONNECTconf
   <3> LT!S_ASTARTreq2
    <4> +response [LT,OUT]                          $Verdict PASS
    <4> LT!S_DATAreq2a
     <5> +response [LT,OUT]                         $Verdict PASS
     <5> LT!S_MINSYNCreq(prn)
      <6> +response [LT,OUT]                        $Verdict PASS
      <6> LT?S_MINSYNCconf(prnc) [prnc=prn] CANCEL(T1;T2)
       <7> {compat := TRUE}                         $Verdict P
        <8> OUT!S_COMPATIBLE(compat)                $Verdict PASS
  <2> LT?OTHERWISE                                  $Verdict FAIL;

$Tree response [LT: K1195; OUT: Disc]
 <1> LT?S_P_EXPTIONind
  <2> LT!S_U_ABORTreq {compat := FALSE}             $Verdict P
   <3> OUT!S_COMPATIBLE(compat)                     $Verdict PASS

$EndTestCase;

$Comment  1988 implementation is tester 1984 implementation is IUT
          According to the ETSI 1988 specifications of FAX4, the attributes
          "Content Architecture Classes" and "Interchange Format Class" can be
          specified in the document profile. This profile is transfered using
          the primitive S_ACT_STARTreq. However, the 1984 implementation can
          not handle such profiles, meaning that problems occur when this
          implementation is the receiver.
          The 1984 implementation will react to this PDU as to a syntactically
          invalid PDU, i.e. it sends a RDGR PDU containing the received PDU up
          to and including the error. At the 1988 implementation this will
          result in a S_P_EXCEPTION5_REPORTind, including the reason and the
          wrong PDU.
$EndComment
```

$TestCase Test_FAX4_84_88_Compatibility/Density_Coding/Case_6
$Purpose Test the different coding of the attribute Pel Transmission Density.
$TestCaseId Dencoding

```
<1> LT!S_CONNECTreq
  <2> LT?S_CONNECTconf
    <3> LT!S_ASTARTreq3
      <4> LT?S_U_ABORTind {compat := FALSE}          $Verdict P
        <5> OUT!S_COMPATIBLE(compat)                 $Verdict PASS
      <4> LT!S_DATAreq2a
        <5> LT?S_U_ABORTind {compat := FALSE}        $Verdict P
          <6> OUT!S_COMPATIBLE(compat)               $Verdict PASS
        <5> LT!S_MINSYNCreq(prn)
          <6> LT?S_U_ABORTind {compat := FALSE}      $Verdict P
            <7> OUT!S_COMPATIBLE(compat)             $Verdict PASS
  <2> LT?OTHERWISE                                   $Verdict FAIL
```

$EndTestCase;

$Comment 1984 implementation is tester 1988 implementation is IUT.
In this testcase the different coding of the attribute "Pel Transmission Density" is tested. In case the 1988 implementation is the receiver, it will react to the relevant PDU as to a syntactically invalid PDU. This means the 1988 implementation will send a S_U_ABORTreq, resulting in a S_U_ABORTind at the tester (= 1984 implementation).
$EndComment

$TestCase Test_FAX4_84_88_Compatibility/Compression_Coding/Case_7
$Purpose Test the different coding of the attribute Compression
$TestCaseId Compcoding

```
<1> LT!S_CONNECTreq
  <2> LT?S_CONNECTconf
    <3> LT!S_ASTARTreq4
      <4> LT?S_U_ABORTind {compat := FALSE}          $Verdict P
        <5> OUT!S_COMPATIBLE(compat)                 $Verdict PASS
      <4> LT!S_DATAreq2a
        <5> LT?S_U_ABORTind {compat := FALSE}        $Verdict P
          <6> OUT!S_COMPATIBLE(compat)               $Verdict PASS
        <5> LT!S_MINSYNCreq(prn)
          <6> LT?S_U_ABORTind {compat := FALSE}      $Verdict P
            <7> OUT!S_COMPATIBLE(compat)             $Verdict PASS
  <2> LT?OTHERWISE                                   $Verdict FAIL
```

$EndTestCase;

$Comment 1984 implementation is tester, 1988 implementation is IUT.
In this testcase the different coding of the attribute "Compression" is tested. In case the 1988 implementation is the receiver, it will react to the relevant PDU as to a syntactically invalid PDU. This means the 1988 implementation will send a S_U_ABORTreq, resulting in a S_U_ABORTind at the tester (= 1984 implementation).
$EndComment

```
$TestStep Test_FAX4_84_88_Compatibility/Session_Resume/preamble
$Purpose  To put the IUT in the state where it expects an
          S_AS_ind after a transmission data error occurred
$TestStepId Preamble [LT: K1195]
 <1> LT!S_CONNECTreq
  <2> LT?S_CONNECTconf
   <3> LT!S_ASTARTreq1
    <4> LT!S_DATAreq2a
     <5> LT!S_MINSYNCreq(prn)
      <6> LT?S_MINSYNCconf(prnc)  [prnc = prn]
       <7> {prn := prn + 1}
        <8> LT!S_DATAFreq
         <9> LT?S_U_EXPTIONind
          <10> LT!S_AINTERreq
            <11> LT?S_AINTERconf
            <11> LT?OTHERWISE              $Verdict FAIL
          <9> LT?OTHERWISE                 $Verdict FAIL
      <6> LT?OTHERWISE                     $Verdict FAIL
  <2> LT?OTHERWISE                         $Verdict FAIL
$EndTestStep

$Comment  1984 implementation is tester, 1988 implementation is IUT.
          This teststep is used to establish a connection between a tester
          (=1984 implementation) and an IUT (=1988 implementation), start the
          exchange of a document, send a wrong data-PDU and start an interrupt
          procedure.
$EndComment

$EndDynamicPart

$EndSuite
```

Appendix F

Contradictions in the 1988 Recommendations

F.1 Contradictions between ODA, ODIF and FAX 4

This section summarizes contradictions between ODA (specified by the CCITT T.410 series Recommendations), ODIF (specified by CCITT Recommendation T.415) and the application profile of facsimile group 4 (specified by CCITT Recommendation T.503).

1. The attribute **content type** is specified as a presentation attribute by ODIF and FAX 4 application profile. ODA specifies the attribute as a content architecture class attribute (see also 5e).
2. The attributes **presentation attributes** can be according to T.412, specified for basic pages, blocks (i.e. basic layout objects) and presentation styles only. According to T.417, they can be specified for basic logical and layout objects, and for presentation styles.
3. The attributes **content architecture class, interchange format class** and **document reference** are classified as mandatory in recommendation T.414 Annex B table 1. Fax 4 application profile does not specify these attributes. The same goes for the S-ACT-START parameter "user data", which is specified in figure 3/T.433 for transparent mode.
4. The attribute **content information** is classified as a raster- graphic coding attribute in T.503. ODIF uses the expression text unit, consisting of content portion attributes (i.e. the attributes "content identifier layout", "type of coding" and "raster-graphic coding attributes") and content information. ODA uses the expression content portion description attributes, consisting of the attributes "content portion identifier", "type of coding", "content information" and "coding attributes". This means that ODA classifies it as an attribute and ODIF classifies it as information which is placed in the information field and not in the attribute field.
5. T.412 (document body) and T.414 (document profile) specify all the attributes of ODA. To specify groups of attributes with common functionalities, T.412 and T.414 use names for these groups. ODIF, in some cases, does not use these names or uses other names.
   These cases are:
   (a) Document constituents attributes (T.414) is not used in ODIF;
   (b) Identification attributes (T.412) is not used in ODIF;
   (c) Relationship attributes (T.412) is not used in ODIF;
   (d) Miscellaneous attributes (T.412) is not used in ODIF;
   (e) Content architecture class attributes (T.412) is not used in ODIF;
   (f) Property attributes (T.412) is not used in ODIF;

(g) Layout object descriptor body is used in ODIF to represent these last five cases. However the attribute "object type" does not belong to this set of attributes (in T.412 it belongs to the identification attributes);

(h) Content information attributes (T.412) is not used in ODIF;

(i) Common coding attributes (T.412) is not used in ODIF.

F.2 Contradictions between DTAM and FAX 4

This section summarizes inconsistencies between DTAM (specified by the CCITT T.430 series Recommendations) and the communication application profile of facsimile group 4 (specified by CCITT Recommendation T.521).

Ambiguities between T.432 and T.521:

1. D-INITIATE:
   (a) The parameter "Telematic Requirements" is conditional in T.432, but mandatory in T.521.
   (b) The parameter "Result" is mandatory in T.432, but a user option in T.521.
2. D-U-ABORT:
   The parameter "User Information" is absent in T.432 but a user option in T.521.
3. D-TRANSFER:
   The parameter "Document Information Type" can have, according to T.432, the values 'transfer completed' or 'transfer not completed' for D-TRANSFER ind or D-TRANSFER conf. According to T.521 this value is always 'transfer a document from its beginning'.
4. D-U-EXCEPTION-REPORT:
   This primitive has according to T.432 no parameters (still under study). T.521 mentions the parameter "User Information" as a user option.

Ambiguities between T.433 and T.521:

1. D-CAPABILITY:
   T.521 mentions the subparameter "Non-Basic Document Characteristics" (like T.432). This subparameter is mapped onto the DCPQ or DCPR APDU. These APDUs have however not such subparameter (the subparameter "Non- Basic Application Characteristics" of these APDUs is the most likely subparameter onto which this service primitive parameter has to be mapped).

2. S-CTRL-GIVE:
   T.433 only uses S-CTRL-GIVE req and S-CTRL-GIVE ind, while T.521 Annex A- 3 also uses S-CTRL-GIVE resp and S-CTRL-GIVE conf.
3. D-U-EXCEPTION-REPORT:
   T.521 specifies this service primitive as applicable to FAX 4 application. T.433 does not use this service primitive.
4. D-P-EXCEPTION-REPORT:
   Like D-U-EXCEPTION-REPORT.
5. S-U-ABORT:
   T.521 specifies in annex A that this service is unconfirmed and that the transport provider initiates a S-U-ABORT ind. primitive at the requesting DTAM-PM. T.433 states that the responding DTAM-PM initiates a D-U-ABORT ind. and a S-U-ABORT resp. after receiving a S-U-ABORT ind. primitive.

Ambiguities between T.432 and T.433:

1. In T.432 the D-CAPABILITY sub-parameter "Document architecture class" can have the values formatted, processable and formatted processable. In T.433 the DCPQ PDU sub-field "Document architecture class" on which the D-CAPABILITY sub-parameter is mapped, can only have the value formatted. Since the D-INITIATE sub-parameter "Document architecture class" and the DINQ PDU sub-field "Document architecture class" can only have as value formatted, it means that only formatted documents can be interchanged by DTAM.

Ambiguities in T.433:

1. Section 6.4.3.2.2. of T.433 states that the responding DTAM-PM issues a S-U-ABORT resp. primitive after receiving a S-U-ABORT ind. primitive. In the state tables in annex B of T.433 S-U-ABORT resp. is not used.

Note: Recommendations T.432 and T.521 use the expression Non Basic Document Characteristics. In Recommendation T.433 this sub-parameter is not used. However comparison of the parameter "Application Capabilities" of D-INITIATE and D-CAPABILITY shows that these have, apart from the D-INITIATE sub-parameter "Non Basic Document Characteristics" and the D-CAPABILITY sub-parameter "Non Basic Application Characteristics", the same sub-parameters. Since the two sub-parameters mentioned have the same definition, it is very likely that they are equal. However, clarification (or adjustment) is necessary.

Bibliography

[1] Weerman, H. *KEMA protocol test service for information technology.* Informatie, vol. 29, no. 10, 1987, pp. 876-9.

[2] Turff, J. *A testing time for OSI.* Communication World Wide (UK), July/Aug 1988, pp. 42-3.

[3] Helwerda, R.J. *Voorverkenningsrapport 4 DNL\89, Conformance test ISDN B-kanaal protocollen.* Internal report PTT Research Neher Laboratorium, jan 1989.

[4] Helwerda, R.J. *Identification of the standards for telematic services on ISDN for the CTS-2-ISDN project. Proposal.* CTS-2- ISDN B-kanaal Work Group, document number CTS-B/PRJ/12 B-ISDN16, status: draft, 21 March 1989.

[5] CCITT Red book, Volume VII - Fascicle VII.3, Terminal Equipment and Protocols for Telematic Services. Recommendations of the T-series, T.0, T.4/Appendix I, T.5, T.6, T.62, T.72, T.73.

[6] Yasuhiko Y., Yamazaki Y., Kamae T. and Kobayashi K. *Advances in FAX.* Proceedings of the IEEE, vol. 73, no. 4, april 1985.

[7] Nemeth K. *Principles of the document interchange protocol for CCITT telematic services.* IEEE Communication magazine, vol. 23, no. 3 march 1985.

[8] Suzuki H. *The CCITT G4 facsimile in the age of ISDNs.* Journal of Electronical Engineering, suppl, 1987.

[9] Hampel H. *High resolution 64 kbit/s facsimile system for ISDN.* Electrical communication: The technical journal of ITT, vol. 60, no. 1, 1986.

[10] Bodson D., Urban S.J., Deutermann A.R. and Clarke C.E. *Measurement of data compression in advanced group 4 facsimile systems.* Proceedings of the IEEE, vol. 73, no. 4, april 1985.

[11] CCITT Blue book, Volume VII - Fascicle VII.3, Terminal Equipment and Protocols for Telematic Services. Recommendations of the T-series, T.411, T.412, T.414, T.415, T.417, T.431, T.503, T.521 and T.563.

[12] CCITT STUDY GROUP VIII - REPORT R2, report of the first meeting of Study group VIII for the study period 1989-1992.

[13] Schulz H.D. and Schumann M. *DTAM Grundlage zukünftiger Telematikanwendungen.* NTZ: NachrichtenTechnische Zeitschrift: Expertwissen über Informationstechnik und Telematik, Bd. 40, Heft 12, 1987, pp. 834-839.

[14] Scheller A. *Document Standards: Availability and Products.* Computer Networks and ISDN Systems, vol 16, 1988, pp. 138-142.

[15] Carr R. *ODA - The ISO Standard for Electronic Document Interchange.* Computer Standards and Interfaces, vol 7, no 3, 1988, pp. 297-301.

[16] Appelt W. *FODA - The Formal Specification of ODA Document Structures and Its Use as a Basis for Conformance Testing.* Computer Standards and Interfaces, vol 7, no 4, 1988, pp. 377-385.

[17] CCITT Recommendation T.62bis.

[18] ETSI Project team 5. Draft NET 31, Testing G4 Facsimile. May 12, 1989.

[19] ISO/IEC JTC 1/SC 21, DIS 9646, *OSI Conformance Testing Methodology and Framework Parts 1, 2 and 3.*

[20] van de Merkhof, P.H.C.A. et. al. *De syntax van RNL- TTCN.* Internal document, no. TTCN/DSS-1/000X, version 1.0, status: Draft, July 7, 1989.

[21] Kleppe, A.G. and van den Merkhof, P.H.C.A. *Syntax and semantics of RNL-TTCN.* Internal document, no. TTCN/DSS-1/0006, version 2.0, status: Draft, May 30, 1989.

[22] Rayner D., *OSI Conformance Testing.* Computer Networks and ISDN Systems, vol 14, 1987, pp. 79-98.

[23] Sarikaya, B. *Conformance testing: Architectures and Test Sequence.* Computer Networks and ISDN Systems, vol 17, 1989, pp. 111-126.

[24] Wang, B. and Hutchinson, D. *Protocol Testing Techniques.* Computer Communications, vol. 10, no. 2, april 1987, pp. 79-87.

[25] Zeng, H.X. and Li, Q. and Du, X.F. and He, C.S. *New advances in Ferry Testing approaches.* Computer Networks and ISDN Systems, vol 15, no. 1, 1988, pp. 47-54.

[26] Helwerda, R.J. *Testing Methodology for the CTS-B Project.* Internal report, Document no.: CTS-B/ITR/2, edition 2, status: draft, 11 April 1989.

[27] CCITT Red book, Volume VIII - Fascicle VIII.7, Data Communication Networks, Message Handling Systems. Recommendations X.408 and X.409.

[28] Tanenbaum, A.S. *Computer Networks.* Second edition. Prentice-Hall International Editions. 1988.