ECMA

EUROPEAN COMPUTER MANUFACTURERS ASSOCIATION

STANDARD ECMA-84

DATA PRESENTATION PROTOCOL

September 1982

Free copies of this document are available from ECMA,
European Computer Manufacturers Association
114 Rue du Rhône – 1204 Geneva (Switzerland)

ECMA

EUROPEAN COMPUTER MANUFACTURERS ASSOCIATION

STANDARD ECMA-84

DATA PRESENTATION PROTOCOL

September 1982

BRIEF HISTORY

This Standard ECMA-84 is one of a set of standards for Open Systems Interconnection (OSI). Open Systems Interconnection standards are intended to facilitate homogeneous interconnection between heterogeneous information processing systems. The standard is within the framework for the co-ordination of standards for Open Systems Interconnection which is defined by ISO/7498.

This ECMA Standard is based on the practical experience of ECMA member companies world-wide, and on the results of their active participation in the current work of ISO and national standard bodies in Europe and the USA. It represents a pragmatic and widely based consensus.

A particular emphasis of this Standard is to specify the homogeneous externally visible and verifiable characteristics needed for interconnection compatibility, while avoiding unnecessary constraints upon, and changes to, the heterogeneous internal design and implementation of the information processing systems to be interconnected.

In the interest of a rapid and effective standardization, the standard is oriented towards urgent and well understood needs. It is intended to be capable of modular extension to cover future developments in technology and needs.

Adopted as Standard ECMA-84 at the General Assembly of June 7-8,1982

TABLE OF CONTENTS

Table of Contents (cont'd)

1 General

1.1 SCOPE

This Standard ECMA-84 for a Data Presentation Protocol:

- defines abstract interactions between two presentation service users via the data presentation service (see Section II),
- defines the protocol to support the above service and its mapping into the underlying session service (see Section III),
- specifies the requirements for conformance with this protocol (see Section IV).

This Standard applies to the Presentation Layer for the Open Systems Interconnection (see ISO/7498). It defines a Presentation Protocol, the main purpose of which is to fully support the requirements of the Virtual File Protocol Standard(ECMA-85). However, it is designed with sufficient generality to be able to also support other (future) Application Layer protocols -standardized or not- with similar requirements, i.e. protocols allowing the exchange between application programs of predictable and repetitive data structures, composed of fields of different data types.

This Data Presentation Protocol is one of several presentation protocols currently defined by ECMA. As such, its definition follows a number of recommendations applicable to all these presentation protocols. At this moment, however, these recommendations are not published as external standards. Therefore, the service and protocol descriptions herein are complete, without reference to any generic standard.

This Standard defines what is needed for compatible interconnection between information processing systems. It does not define local interactions between a presentation service user and the presentation service. It in no way defines inter-layer interfaces.

1.2 REFERENCES

ISO/7498	Open Systems Interconnection - Basic Reference Model.
ECMA-6	7-bit Input/Output Coded Character Set.
ECMA-43	8-bit Coded Character Set.
ECMA-75	Session Protocol.
ECMA-85	Virtual File Protocol.

1.3 GENERAL OVERVIEW

The main role of the data presentation service is to allow a meaningful exchange of information between two application entities, by allowing the negotiation of a standard Transfer Environment defining the agreed upon representation of application information during its transfer. Any mapping between this agreed upon representation and specific local representations at either systems is a responsibility of the implementations and not defined in this Standard.

Another role of the data presentation service is to make available to its user the dialogue services offered by the underlying session services. These services are not transformed by the Presentation Layer, which in the case of the data presentation service assumes no responsibility in the dialogue discipline. Therefore, their description is entirely by reference to the Session Protocol Standard (ECMA-75).

2 Service

2.1 SERVICE OVERVIEW

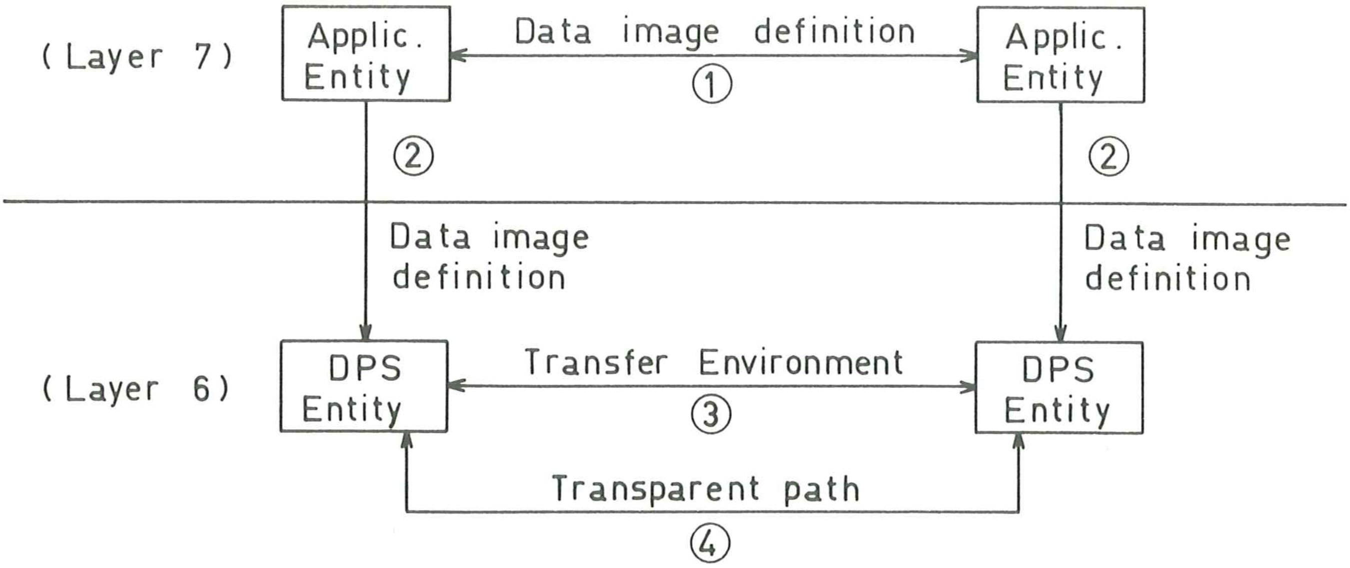
2.1.1 Relationship between the Application Layer and the Presentation Layer

The data exchanged between two application entities can be characterized in two ways:

- an abstract way (the data image definition),
- a syntactic way (the Transfer Environment).

This subclause models the respective views of data of the Application and Presentation Layers and the relationships between these views. This is illustrated in Fig. 1.

Fig. 1 - Application and Presentation views of data



2.1.1.1 Data image definition

The data image definition is an abstract description of application data. As such, it has to be known by both application entities (DPS users) and the way the data image definition is exchanged between these two entities is considered to be a responsibility of the Application

Layer protocol (see arrow 1 in Fig. 1). Therefore, the DPS offers no facility for data image definition exchange.

It is left to each local implementation to decide if and how the data image definition has to be communicated by the DPS user to the DPS entity (See arrow 2 in Fig. 1): this depends on how the syntax mapping is implemented.

A data image definition typically consists of an ordered list of field descriptions, where each field is characterized by its data type and size or range constraints. There are however simple situations, where the data is fully homogeneous and therefore the data image definition consists only of a single data type.

The data image definition can be changed during the life of an application connection. There is only one current image definition at any time.

2.1.1.2 Transfer Environment

The Transfer Environment (See arrow 3 in Fig. 1) is composed of:

- an agreed upon set of data types;
- the syntax used for representation of each of the valid data types during their transfer between two presentation entities;
- the optional use of a compressed form of data during the transfer.

A transparent path allows to exchange data between DPS users independently of the established Transfer Environment (See arrow 4 in Fig.1).

2.1.1.3 Data types

Data types are the common elements that relate data image definition and data syntax. There are two major data types supported by this Standard: strings and numbers.

Strings are further subdivided according to their elements: character strings, octet strings (for transparent data) and bit strings. For each of these, the size constraint is expressed in number of elements in the string. A given string field in the data image definition is always of fixed size in all its occurences in this version. The syntax attributes are:

- character code for character strings,
- alignment rule for bit strings: packed (each bit string field starts at the first bit following the previous field) or unpacked (each bit string field starts at an octet boundary). Fields which are not bit strings always start at an octet boundary.

Numbers are further characterized by a scale (fixed or floating point), a base (binary digits or decimal digits) and a sign option (signed or unsigned). For each of these the range constraint is expressed in number of digits (binary or decimal) to represent the maximum possible value of the field. In addition, the number of digits in the exponent is indicated for floating-point numbers. The syntax attributes are:

- packing option for decimal numbers: packed (two digits per octet) or unpacked (one digit per octet).
- for binary numbers, no attributes are defined in this version: that is, a single standard syntax is offered.

NOTE 1

For fixed-point numbers, it is assumed that the location of the fixed point is known only by the users.

NOTE 2

Floating-point decimal numbers and unsigned floating-point numbers are excluded.

NOTE 3

The base attribute for numbers is considered part of the data type because it influences the performance of the application programs: users select the base according to the most frequent type of processing of the field (editing or arithmetic).

2.1.1.4 Transfer Environment negotiation

The purpose of the Transfer Environment (TE) negotiation is to (re-) negotiate the usage of the various data types, their current syntax and the compression method used. A given TE is established by a single interaction, starting from the current environment: all items not explicitly negotiated retain their current definition.

The initial TE is composed of the following data types and their associated syntaxes:

- character strings encoded according to standard ECMA-6
- octet strings.

Furthermore the usage of the standard compression method (called STD-1) is negotiated.

The following rules apply to the TE negotiation:

1. The initiator proposes a set of data types $\{DT_1,...,DT_n\}$ $(n>0, DT_i \neq DT_j$ for all $i \neq j)$ for the next TE, i.e. to

become valid after the current negotiation. For each data type, the initiator also proposes a set of data syntaxes $\{DSi_1,\ldots,DSi_m\}$ $(i_m > 0)$ in the order of preference. During TE negotiation, the initiator may also propose a set of compression methods $\{C_1,\ldots,C_\ell\}$ $(\ell \geq 0)$ it is able to support.

2. The acceptor responds with all data types it supports: $0 \subseteq \{dt_1,\ldots,dt_k\} \subseteq \{DT_1,\ldots,DT_n\}$ $(0 \leq k \leq n,\ dt_i \neq dt_j$ for all $i \neq j)$.
   It also selects one of the proposed data syntaxes $ds \in \{DSi_1,\ldots,DSi_m\}$ for each $dti \in \{dt_1,\ldots,dt_k\}$.
   For each supported data type, such a syntax exists by means of the definition of a mandatorily supported data syntax for each data type. It should however be noted that support of all data types is optional, except octet and character string.

   If proposed by the initiator, the acceptor also selects one or none of the proposed compression methods. None means that no compression is defined in the TE.

2.1.2 Summary description of services

2.1.2.1 Connection facility

The connection facility provides for establishment and release of the presentation connection.

At connection establishment, there is a negotiation of the presentation service subset to be used. An initial Transfer Environment is also established.

Connection termination can be requested by either user. Both orderly and forced termination are provided. The connection can also be accidentally lost: this is reported to both users by the presentation service.

2.1.2.2 Presentation facility

The presentation facility provides all services related to data presentation. This includes transfer of application data according to the current Transfer Environment and re-negotiation of the Transfer Environment. Re-negotiation is available only if the proper service subset has been agreed at connection establishment. If re-negotiation is not part of the agreed service subset, the initial Transfer Environment is used for the entire duration of the presentation connection. Else, the Transfer Environment can be re-negotiated by use of the single interaction negotiation enclosure.

A transfer Environment defines the set of data types supported by both presentation entities and their agreed representation; it also defines the agreed data compression method, if any. Re-negotiation of the Transfer

Environment is not necessary every time that the data image definition changes, unless the newly defined data image contains data types for which no support is provided in the current Transfer Environment.

2.1.2.3 Dialogue facility

This includes a number of services to aid the control of dialogue between two application entities. They are directly derived from session services, without any change of functionality. This includes synchronization facilities and token management. The Data Presentation Protocol by itself makes a very limited use of the session services for its own purpose and in particular uses none of the synchronization and token management services: this allows the application entities to fully monitor their dialogue.

2.1.3 List of services

Table 1 below lists all the services of the Data Presentation Protocol. For each service, it specifies its type (See Appendix C) and its purpose.

Table 1 - Data presentation services

Service	Type	Description
		CONNECTION FACILITY
P-CONNECT	2	Establish DPS connection
P-RELEASE	2	Orderly release of DPS connection
P-DISCONNECT	1	Forced release of DPS connection
P-ABORT	3	Loss of DPS connection
		PRESENTATION FACILITY
P-PERFORM-NEGOT.	2	Re-negotiate Transfer Environment
P-DATA	1	Transfer DPS user data
		DIALOGUE FACILITY
P-SYNC	2	Set minor synchronization point
P-END-DU	2	Set major synchronization point
P-RESYNC	2	Reset the session connection
P-PLEASE	1	Request for token(s)
P-TOKENS-GIVE	1	Passing of token(s)

2.2 SERVICE DESCRIPTION

2.2.1 Primitives

The service is described by using the service description notation and terminology defined in Appendix C.

2.2.1.1 The P-CONNECT service

Purpose

To establish a presentation connection between the initiating DPS-user and a peer DPS-user.

Structure

Confirmed, type 2.

Parameters

	REQ	IND	RESP	CONF
Diagnostic	x	x	B	U
User-caller	D	U	x	x
User-called	D	U	D	U
Service-ident	D	U	x	x
Service-version	D	U	D	U
Service-subset	D	U	D	U
Transp-data	D	U	D	U
Initial-te-parameters				
Compression	D	U	D	U
Initial-service-parameters				
Special-conventions	D	U	D	U
Session-subset	D	U	x	x
Other-session-parameters	D,x	U,x	D,x	U,x

Parameter description

Diagnostic

Advises the result of the service element. See 2.2.2 for further information.

User-caller

Optional; if present, gives the name of the caller and is passed in the indication.

User-called

The name of the called DPS-user. Optional in response and confirmation primitives.

Service-ident

Applicable value is: DPS.

Service-version

Designates the DPS version. Negotiable. Applicable value for this version is: 1.

Service-subset

Applicable values are: DP-A, DP-B. Negotiable; the response primitive can contain DP-B only if the indication primitive contained DP-B. See 2.3.

Transp-data

Optional; information is supplied by the DPS-user and conveyed transparently by the DPS. Maximum length is 45 octets.

Compression

Enables the use of the standard compression algorithm STD-1 (See 3.3.2.9). Negotiable. Applicable values are: on and off. The response primitive can contain on only if the indication primitive contained on.

Special-conventions

Optional; specifies which special presentation conventions may be used on this connection. Negotiable. Successful negotiation allows the use of the special-information parameter and of the "private values" where these apply. Else, use of these facilities is illegal. See Appendix E for purpose of defining special conventions.

Session-subset

See ECMA-75 for the definition of this parameter. The only values applicable here are: C and D. The value selected by the DPS-user is directly passed to the session service. For max-ssdu-sizes, the minimum value should be 82 octets.

Other-session-parameters

Include session service parameters of the S-CONNECT applicable to the selected session-subset (See ECMA-75). The values selected by the DPS-users are directly passed to the session service.

Effects

This service element is sequenced. No further request primitive will be accepted from the initiating DPS-user until the confirmation primitive has been issued, except P-DISCONNECT request primitive. No primitive from the acceptor will be allowed to overtake the confirmation except P-DISCONNECT which will destroy the effects of P-CONNECT and of any other issued request primitive. A P-ABORT will disrupt a P-CONNECT action and any other service element in progress.

Additional information

User-caller and user-called will not necessarily retain their values unchanged between the two DPS-users.

2.2.1.2 The P-RELEASE service

Purpose

To obtain an orderly release of a presentation connection. To issue a P-RELEASE, the DPS-user must hold the data token and the terminate token if these tokens have been defined in the session parameters of P-CONNECT.

Structure

Confirmed, type 2.

Parameters

	REQ	IND	RESP	CONF
Diagnostic	x	x	D	U
Transp-data	D	U	D	U

Parameter description

Diagnostic

Advises the result of the service element. See 2.2.2 for further information.

Transp-data

Optional; information is supplied by the DPS-user and conveyed transparently by the DPS. Maximum length is 24 octets.

Effects

This service element is sequenced and non destructive. No other request primitive (except P-DISCONNECT) will be accepted from the request issuer until the response/ confirmation has been given, but further indication primitives may be given to the request issuer due to service elements initiated by the peer DPS-user before the P-RELEASE indication primitive was given to the peer DPS-user.

Additional information

If the diagnostic severity is "failure", the previous state of the DPS connection is restored and requests will again be accepted according to the normal rules of the DPS. This can apply only if the session terminate token is used to mediate the issue of P-RELEASE request primitive. In the case of "success", the session connection is also broken.

2.2.1.3 The P-DISCONNECT service

Purpose

To force termination of a presentation connection. May be issued by either DPS-user at any time.

Structure

Non-confirmed, type 1.

Parameters

	REQ	IND
Transp-data	D	U

Parameter descriptions

Transp-data

Optional; information supplied by the DPS-user is conveyed transparently by the DPS. Maximum length is 3 octets, the first octet will have the most significant bit forced to ZERO.

Effects

This service element may be expedited and disruptive. The DPS connection is broken and the session connection disconnected also.

Additional information

In the event of a collision of P-DISCONNECT request primitives, either or both indication primitives may be lost.

2.2.1.4 The P-ABORT Service

Purpose

Generated by the DPS to indicate that a DPS connection has been lost due to DPS or lower layer service exception condition. May be generated at any time.

Structure

Indication only, type 3.

Parameters

	IND	IND
Diagnostic	U	U

Parameter description

Diagnostic

Indicates the reason for loss of the connection. See 2.2.2 for further information. A different value may be provided to the two DPS-users in some cases.

Effects

This service element is disruptive. The DPS connection has been broken and the session connection is disconnected if this is not already lost (by S-ABORT). All parameters of the DPS connection are assumed lost, and no further indications or confirmations will be received for this DPS connection.

2.2.1.5 The P-PERFORM-NEGOTIATION service

Purpose

To re-negotiate the transfer environment. The request primitive may be issued only by the DPS-user who currently holds all defined session tokens, except the terminate token.

NOTE 4

Since only subsets C and D of session service are supported, there is always at least one such token defined and no collision will therefore happen.

Structure

Confirmed, type 2.

Parameters

	REQ	IND	RESP	CONF
Diagnostic	x	x	B	U
Te-specific-parameters				
Data-type-definition	D	U	D	U
Data-type-syntax	D	U	D	U
Compression-method	D	U	D	U
Special-information	D	U	D	U

Parameter description

Diagnostic

Advises the result of the service element. See 2.2.2 for further information.

Data-type-definition

Values are: character string, bit string, octet string, unsigned fixed binary, signed fixed binary, unsigned fixed decimal, signed fixed decimal, signed floating point. See 2.1.1.4 for negotiation rules.

Data-type-syntax

Applicable values depend on the associated data-type-definition value. For character string: ECMA-6, ECMA-43. For bit string or fixed decimal: unpacked, packed. In addition, private values are reserved for all data types. See 2.1.1.4 for negotiation rules.

Compression-method

Specifies if a compression algorithm is to be used. Applicable values are: none, STD-1. See 2.1.1.4 for negotiation rules and 3.3.2.9 for description of STD-1 algorithm.

Special-information

Special information about the transfer environment conforming to the special-conventions agreed at establishment of the connection.

Effects

This service element is sequenced and non destructive. The request issuer is not permitted to issue any further requests (except P-DISCONNECT) until the confirmation is received, but must be prepared to receive further indications. The peer DPS-user is not permitted to issue further requests (except P-DISCONNECT) after receipt of the indication until the response is sent.

If negotiation is successful, the new Transfer Environment is established. Else, the former Transfer Environment is reestablished, as though the request had never been issued.

2.2.1.6 The P-DATA service

Purpose

To transfer presentation user data formatted according to the current Transfer Environment and/or transparent user data. If a session data token is defined, only the entity holding the data token may issue a P-DATA.

Structure

Not confirmed, type 1.

Parameters

	REQ	IND
Formatted-data	D	U
Transp-data	D	U

Parameter description

Formatted-data

User data subject to syntax transformation and compression in the current Transfer Environment.

Transp-data

Transparent presentation user data, not subject to any transformation in the current Transfer Environment.

Effects

This service element is sequenced and non destructive.

2.2.1.7 The DP-SYNC service

See S-SYNC in ECMA-75. All parameter values are passed to the session service as set by the DPS-user. The DPS imposes that only "normal" synchronization points can be defined.

2.2.1.8 The DP-END-DU service

See S-END-DU in ECMA-75. All parameter values are passed to the session service as set by the DPS-user.

2.2.1.9 The DP-RESYNC service

See S-RESYNC in ECMA-75. All parameter values are passed to the session service as set by the DPS-user. An additional constraint on the usage of S-RESYNC is imposed by the DPS: the Transfer Environment is unaffected by a re-synchronization, i.e. a request for a re-synchronization to a point within a previous Transfer Environment is invalid.

2.2.1.10 The DP-TOKENS-GIVE service

See S-TOKENS-GIVE in ECMA-75. All parameter values are passed to the session service as set by the DPS-user.

2.2.1.11 The DP-PLEASE service

See S-PLEASE in ECMA-75. All parameter values are passed to the session service as set by the DPS-user.

NOTE 5

The availability of the services described in 2.2.1.7 to 2.2.1.11 depends on the session subset selected in P-CONNECT.

2.2.2 Error reporting

Error reporting is provided by means of a diagnostic parameter appearing in response and confirmation primitives. It also appears in a few request and indication primitives (for disruptive services). The diagnostic parameter conveys up to three elements of information, corresponding to three levels of error analysis:

- severity
- reason
- diagnostic supplement

Each element can be supplied only if the preceding (more synthetic) elements have been supplied.

2.2.2.1 Severity

Specifies the degree of success or failure. The legal values are, by increasing order of severity:

- Success
- Success with warning
- Failure

Severity is mandatory in a diagnostic parameter.

2.2.2.2 Reason

Specifies a summary diagnostic. Legal values are supplied in 3.3.2.4.

2.2.2.3 Diagnostic supplement

Specifies a detailed diagnostic. Several diagnostic supplements may be specified within the same diagnostic parameter. The legal types of diagnostic supplements are defined below.

DS-0: May accompany any reason value and is specifically expected when the reason value is "non standard reason". The legal value is a character string of not more than 31 characters.

DS-1: Accompanies reason values like "unset parameter value", "illegally duplicated parameter", "illegal parameter value". Repeated once for each erroneous parameter. Contains the parameter type of the defective parameter.

DS-2: Accompanies reason values like "protocol violation". Contains the type of the erroneous message.

2.3 SERVICE SUBSETS

2.3.1 General

Subsets of the data presentation service are defined to achieve simplification and variety control. The following subsets are applicable for the data presentation service:

1. No re-negotiation (subset DP-A);

2. Re-negotiation by single interaction negotiation (subset DP-B).

The supported dialogue facility services depend only on the session subset choice in P-CONNECT: either subset C or subset D (see ECMA-75 for definition of these subsets).

2.3.2 Subset DP-A

This subset is composed of all the services listed in 2.1 of this standard, except the P-PERFORM-NEGOTIATION service.

It allows for the handling of only two data types, the support of which is mandatory:

- octet strings;
- character strings (encoded according to ECMA-6).

The compression algorithm described in this Standard (See 3.3.2.9) is included in this subset. Its support is optional.

2.3.3 Subset DP-B

This subset is composed of all the services listed in 2.1 of this Standard, including the P-PERFORM-NEGOTIATION service.

All data types and data type representations defined in this Standard are applicable, but their support is optional except for those defined in subset DP-A. For each data type supported, one syntax is mandatory, the others being optional: this insures that two presentation entities that support a given data type always have at least one common syntax for this data type. The preferred (mandatory) syntaxes are:

- character string: encoded according to ECMA-6
- bit string: unpacked
- fixed decimal: unpacked.

The compression algorithm described in this Standard (See 3.3.2.9) is included in this subset. Its support is optional.

NOTE 6

Other compression algorithms may be added in future versions of this standard. The one described here (referred to as STD-1) will remain the preferred algorithm: that is, if compression is supported, the compression method designated by STD-1 must be supported: this is to insure that two presentation entities that want to support compression always have at least one common compression method.

3 Protocol

3.1 PROTOCOL OVERVIEW

3.1.1 Descriptive model

The Data Presentation Protocol (DPP) is modelled as an abstract machine, with protocol structures between the two DPS entities. A protocol structure is an elementary dialogue for the purpose of an indivisible operation. As such, it is totally successful or totally unsuccessful, never partly successful. It is composed of a request, issued by one DPS entity, and for some types of structures of a response, issued by the other DPS entity. Each response or request is composed of a single protocol message.

There are two types of protocol structures:

- Type 1 structure: request without response
- Type 2 structure: request with response

A protocol message contains protocol control information (i.e.one or more parameters) and may in some cases also contain user data.

Dynamic execution of the DPP results in an ordered sequence of protocol structures. To describe the protocol, it is sufficient to separately describe each of its structures (or messages), plus any precedence relationship between structures (state transitions).

3.1.2 List of protocol structures

Table 2 below lists all the structures of the DPP. For each structure, it specifies its type (See 3.1.1) and its purpose.

Table 2 - Protocol structures

Structure	Type	Description
PP-CONNECT	2	Initiate DPS connection
PP-RELEASE	2	Orderly release of DPS connection
PP-DISCONNECT	1	Forced release of DPS connection
PP-PERFORM-NEGOT.	2	Re-negotiate Transfer Environment
PP-DATA	1	Transfer user data

NOTE 7

The DPP messages for the dialogue facility are not described here. Refer to 3.4.6 and ECMA-75.

3.2 PROTOCOL DESCRIPTION

3.2.1 Notation

This clause provides a narrative description of the protocol. Appendix D provides the formal description.

Each message is defined by the following items:

- function,
- list of parameters,
- relationship with service primitives (sending/receiving).

A detailed description of parameters is supplied only when they differ from the service parameters; otherwise, reference is made to the description of the equivalent parameter in the service.

3.2.2 PP-CONNECT-RQ

Function: Initiate a DPS connection.

```
Content:  User-caller
          User-called
          Protocol-definition = {service-ident
                                {service-version
                                {service-subset
          Transp-data
          Compression
          Special-conventions
          Session-subset
          Other-session-parameters
```

Parameter description: See P-CONNECT, 2.2.1.1.

Sending: On P-CONNECT request primitive.

Receiving: Generates a P-CONNECT indication primitive. The expected outcome is a P-CONNECT response primitive.

3.2.3 PP-CONNECT-RP

Function: Response to PP-CONNECT-RQ.

```
Content:  Diagnostic
          User-called
          Protocol-definition = {service-version
                                {service-subset
          Compression
          Transp-data
          Special-conventions
          Other-session-parameters
```

Parameter description: See P-CONNECT, 2.2.1.1.

Sending: On P-CONNECT response primitive.

Receiving: Generates a P-CONNECT confirmation primitive.

3.2.4 PP-RELEASE-RQ

Function: Request orderly release of DPS connection.

Content: Transp-data.

Parameter description: See P-RELEASE, 2.2.1.2.

Sending: On P-RELEASE request primitive.

Receiving: Generates a P-RELEASE indication primitive. The expected outcome is a P-RELEASE response primitive.

3.2.5 PP-RELEASE-RP

Function: Response to PP-RELEASE-RQ.

Content: Diagnostic
Transp-data

Parameter description: See P-RELEASE, 2.2.1.2.

Sending: On P-RELEASE response primitive.

Receiving: Generates a P-RELEASE confirmation primitive.

3.2.6 PP-DISCONNECT-RQ

Function: Request forced release of DPS connection.

Content: Transp-data or Diagnostic

Parameter description: See P-DISCONNECT, 2.2.1.3 for transp-data and P-ABORT, 2.2.1.4 for diagnostic.

Sending: On P-DISCONNECT request primitive or when DPS entity detects the need to abort the connection.

Receiving: Generates a P-DISCONNECT indication primitive or a P-ABORT indication primitive, according to the parameter which is present.

3.2.7 PP-PERFORM-NEGOTIATION-RQ

Function: Re-negotiate Transfer Environment.

Content:
```
TE-specific-params = {data-type-definition
                     {data-type-syntax
                     {compression-method
Special information
```

Parameter description: See P-PERFORM-NEGOTIATION, 2.2.1.5.

Sending: On P-PERFORM-NEGOTIATION request primitive.

Receiving: Generates a P-PERFORM-NEGOTIATION indication primitive. The expected outcome is a P-PERFORM-NEGOTIATION response primitive.

3.2.8 PP-PERFORM-NEGOTIATION-RP

Function: Response to PP-PERFORM-NEGOTIATION-RQ

Content:
```
Diagnostic
TE-Specific-parameter = {data-type-definition
                        {data-type-syntax
                        {compression-method
Special-information
```

Parameter description: See P-PERFORM-NEGOTIATION, 2.2.1.5.

Sending: On P-PERFORM-NEGOTIATION response primitive.

Receiving: Generates a P-PERFORM-NEGOTIATION confirmation primitive.

3.2.9 PP-DATA-RQ

Function: Transfer user data.

Content: Formatted-data
Transp-data

Parameter description: See P-DATA, 2.2.1.6.

Sending: On P-DATA request primitive.

Receiving: Generates a P-DATA indication primitive.

3.3 PROTOCOL ENCODING

General convention: the bits of an octet are identified b1, b2, b3, b4, b5, b6, b7, b8, b1 being the leftmost and most-significant bit. The same convention applies to strings of more than one octet, the rightmost bit becoming b16, b24 or b32.

3.3.1 Message structure

Each message of the DPP is composed of two parts:

- a one-octet message header,
- a variable-length message content.

The message header contains an 8-bit message type. The legal type values are listed in Table 3.

The message content is composed of the message parameters, in any order. The representation of parameters is described in 3.3.2.

Table 3 - Message type values

1:	PP-CONNECT-RQ
2:	PP-CONNECT-RP
3:	PP-RELEASE-RQ
4:	PP-RELEASE-RP
5-7:	not assinged
8:	PP-PERFORM-NEGOTIATION-RQ
9:	PP-PERFORM-NEGOTIATION-RP
10-63:	not assigned
64-70:	PP-DATA-RQ (*)
71-255:	not assigned

(*) The rules for assignment of the message type of a given PP-DATA-RQ message are given in 3.3.2.7.

The PP-DISCONNECT-RQ message is directly mapped onto a session service and its user-data parameter. The message type does not explicitly appear in this version.

3.3.2. Parameter encoding

3.3.2.1 Type/Length/Value technique (TLV technique)

The TLV technique is a method for coding an Information Unit.

Every information unit is encoded as a triplet, made of:

- a type (1st field),
- a length (2nd field),
- a value (3rd field).

Type field (1 octet)

b1 = 0

b2-b8 = type value (1 to 127, 0 reserved).

Length field (1 or 2 octets)

b1 = 0: the length is specified on 7 bits (b2-b8)
b1 = 1: the length is specified on 15 bits (b2-b16)

b2 to b8 or to b16: number of octets of the value field expressed in binary notation.

Value field (0 to N octets)

All octets: data.

3.3.2.2 Parameter value representation

The following value types are needed for representation of the various parameters of the DPP:

C: Character string. Any size (unless limits are defined), character encoding is according to the ECMA 7-Bit Coded Character Set (see ECMA-6). Each character is mapped on bits b2-b8 of an octet, with bit b1 equal to ZERO. The allowed characters are those in columns 2 to 7 of the International Reference Version, with the exclusions of those in positions: 4/0, 5/11 to 5/14, 6/0, 7/11 to 7/15.

N: Numeric. Unsigned binary integer, with four possible sizes: 8, 16, 24 or 32 bits.

S: Symbolic. Unsigned binary integer, the values of which encode specific meanings. Size is 8 bits. Values to which no meaning has been assigned are reserved and should not be used.

M: Bit map. Bit string in which each bit encodes a specific meaning. Size is 8 or 16 bits. Any bit the role of which is not defined must be encoded as ZERO. For bits representing specified options, the bit is set to ONE if the option is requested, ZERO otherwise.

T: Transparent. Encoded by DPS user.

A: Aggregate. Composed of fields (sometimes smaller than 8 bits) of the above types.

Value truncation

For economy reasons, it is recommended to use the minimum variable length strings for expressing values.

- character strings should not contain unnecessary spaces,
- for numeric or symbolic, all unnecessary octets from left containing only ZEROs should be removed,
- for bit maps, all unnecessary octets from right containing only ZEROs should be removed (must be from right to allow future extension to the right),
- for transparent, no truncation is applied.

3.3.2.3 Parameter encoding

For each parameter of the protocol, Table 4 gives the following description:

- parameter type (unique identifier,)
- maximum length of value field (- : means unlimited),
- type of value representation (C,N,S,M,T : see 3.3.2.2),
- encoding of all possible values (for S or M only).

The parameters which are directly mapped onto session service parameters do not appear in this Table.

Default values

The existence of a default value, to be used in case the parameter has not been explicitly specified, depends only on the parameter value type (See 3.3.2.2):

- for value types S and M, there is always a default value, equal to ZERO,
- for value types C, N and T, there is no default value.

Aggregates

A few parameters contain, instead of an elementary value, an aggregate of values. This aggregate is encoded as a fixed data structure when most elements are always present and there is no need for extendability. Otherwise, it is encoded recursively as a set of TLV items. In this latter case, the range of Ts internal to an aggregate can overlap the range of Ts defined in Table 4, since this represents another level of encoding.

The internal encoding of aggregate parameters is specified in 3.3.2.4 to 3.3.2.6.

Table 4 - Parameter encoding

Type	Parameter	Value length	Value type	Value encoding
1	Diagnostic	-	-	Aggregate, see 3.3.2.4
2	Transp-data	-	T	See special cases below
3	Protocol-definition	3	-	Aggregate, see 3.3.2.5
10	TE-specific-parameter	-	-	Aggregate, see 3.3.2.6
22	Special-conventions	2	T	
23	Special-information	-	T	
64	Compression	1	S	1: STD-1 algorithm 0: no compression
-	Formatted-data	-	-	See 3.3.2.7 and 3.3.2.8

Special cases for transp-data:

The transp-data parameter is encoded without the TL items in two cases where an extremely efficient encoding is required:

- In the PP-DISCONNECT-RQ (because only 3 octets for the entire parameter are available, due to session mapping);
- in the PP-DATA-RQ messages (See 3.3.2.7).

3.3.2.4 Value of the diagnostic parameter

The value field of the diagnostic parameter is a structure defined as follows in BNF:

<diagnostic value> ::= <severity> <reason> {<DS>}

<DS> ::= <DS type> <DS value length> <DS value>

The terminal elements are encoded as follows:

<severity>

4-bit binary integer (b1-b4 of first octet of <diagnostic value>). See Table 5 for values. When the diagnostic parameter is omitted, the default value of <severity> is ZERO (success).

<reason>

12-bit binary integer (b5-b8 of first octet of <diagnostic value> followed by b1-b8 of second octet). See Table 6 for values. When the diagnostic parameter is omitted, the default value of <reason> is ZERO (no reason).

<DS type>

3-bit binary integer (b1-b3 of first octet of <DS>). Legal values: 0 to 2.

<DS value length>

5-bit binary integer (b4-b8 of first octet of <DS>).

<DS value>

Dependent on <DS type> value, as follows:

0: character string, up to 31 characters.

1: one octet containing a parameter type (See 3.3.2.1)

2: one octet containing a message type (See 3.3.1)

Table 5 - Severity values

0: success
1: success with warning
2: (reserved)
3: failure

Table 6 - Reason values

Value	Sev.	Reason
0	0	No reason provided
1	1,2,3	Non standard reason
2	3	Unset parameter value
3	3	Illegal parameter value
4	3	Unsupported parameter value
5	3	Illegally duplicated parameter
6	3	Illegal parameter type
7	3	Unsupported parameter type
12	3	Protocol sequence violation
19	3	Illegally duplicated parameter value
20	3	Conflicting parameter value

Encoding of parameter to PP-DISC-RQ

This is treated as a special case since only three octets are available in the ss-user-data parameter to S-DISCONNECT into which this message is mapped, and two forms of parameter are required in the DPP.

The encoding in this case is as follows:

<DNQ param> ::= <designator><transp-data>/<special diagnostic>

<special diagnostic> ::= <s-severity><reason><s-DS-value>

The terminal elements are encoded as follows:

<designator>
A single bit; b1 of first octet of parameter. Values are as follows:

- 0: transp-data is present
- 1: special diagnostic is present

<transp-data>
Three octets of information from transp-data; the first octet, truncated to bits b2-b8, is encoded into bits b2-b8 of the first octet and the remaining two octets are encoded into the remaining two octets.

<s-severity>
This is similar to <severity> in 3.3.2.4, but is a 3-bit binary integer encoded in b2-b4 of the first octet.

<reason>
As in 3.3.2.4.

<s-DS-value>
Single octet conforming to definition of <DS-value> in 3.3.2.4 for DS type 2 and encoded into the third octet.

3.3.2.5 Content of protocol-definition aggregate

Parameters of this aggregate are encoded as a structure containing a fixed number of elements. This structure is described by Table 7.

Table 7 - Protocol-definition encoding

Type	Parameter	Value length	Value type	Value encoding
-	Service-identifier	1	S	1: DPP
-	Service-version	1	M	b1: version 1
-	Service-subset	1	M	b1: subset DP-B

3.3.2.6 Content of TE-specific-parameters aggregate

One TE-specific-parameters aggregate may contain exactly one data-type-definition parameter and ZERO (i.e. the default data type syntax, which is equivalent to the ZERO value is assumed), or one or more data-type-syntax parameters. The values of the data-type-syntax parameters are to be interpreted according to the value of the associated data-type-definition parameter. Alternatively, a TE-specific-parameters aggregate may contain a compression-method parameter.

Different TE-specific-parameters aggregates in the same message shall not contain the same value of a data-type-definition parameter or a second compression-method parameter. The rules for use of the TE-specific-parameters aggregates within the PP-PERFORM-NEG-RQ/RP are specified in 2.1.1.4.

Table 8 - TE-specific-parameters encoding

Type	Parameter	Value length	Value type	Value encoding
1	Data-type-definition	1	A	b1-b2: major type: symbolic 0: string, 1: number b3-b8: attributes - for type=0, b3: size type (0:fixed) b4-b6: reserved (0) b7-b8: element type (0: character, 1: octet, 2: bit) - for type=1, bit map b3-b5: reserved b6: base 0: bin, 1: dec (0 for floating point) b7: scale 0:fix, 1:float b8: sign (0: unsigned 1: signed)
2	Data-type-syntax	1	S	- for character string: 0: ECMA-6 1: ECMA-43 - for bit strings and fixed decimal numbers: 0: unpacked 1: packed - for all data types: 192-255: private encoding
3	Compression-method	1	M	b1: STD-1, see 3.3.2.9

The standard representations for the various data types are described in 3.3.2.8.

3.3.2.7 Formatted and transparent data encoding in PP-DATA-RQ

The formatted-data and transp-data parameters can appear in the PP-DATA-RQ message. For maximum efficiency, they are not encoded in TLV form and the message header is generally only one octet: this is obtained by defining different message types for the PP-DATA-RQ, according to the respective amount of formatted and transparent data and whether compression is applied. Transparent data, when present, always precedes formatted data.

Transparent data only:

TYPE = 64	Transp-data

Formatted data only (compressed):

TYPE = 65	Formatted + compressed data

Formatted data only:

TYPE = 66	Formatted data

Formatted data (compressed) and one octet of transparent data:

TYPE = 67	Transp-data	Formatted + compressed data

Formatted data and one octet of transparent data:

TYPE = 68	Transp-data	Formatted data

Formatted data (compressed) and N octets of transparent data:

TYPE = 69	transp-data-length	Transp-data	Formatted + compressed data

Formatted data and N octets of transparent data:

TYPE = 70	transp-data-length	Transp-data	Formatted-data

(transp-data-length is encoded on one or two octets, in the same way as the length field of a TLV. See 3.3.2.1).

3.3.2.8 Data types representations

OCTET STRING:

Octets, as encoded by the DPS user.

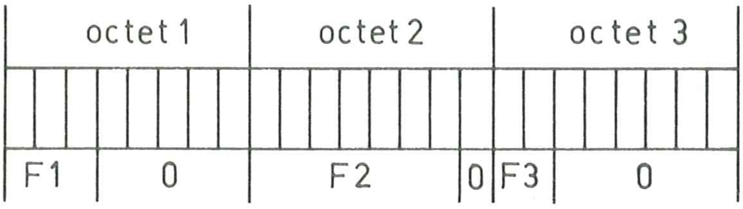
CHARACTER STRING:

One character per octet, right-justified in the octet, with ZEROs as padding bits if necessary.

BIT STRING (UNPACKED):

Series of bits left-justified into the minimum number of octets, with ZEROs as padding bits if necessary. Each bit string starts with b1 of an octet.

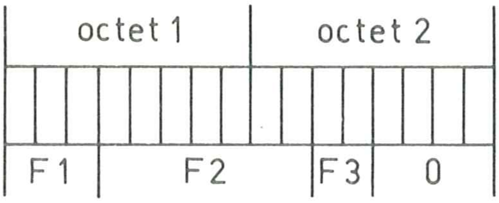
Example for three consecutive bit string fields (respectively 3, 7 and 2 bits); these fields are assumed to be preceded and followed by aligned fields (e.g. character strings).



BIT STRING (PACKED):

Series of bits starting at the first bit position after the preceding field. When the preceding field is also a bit string, the first bit position is not necessarily bit b1 of an octet.

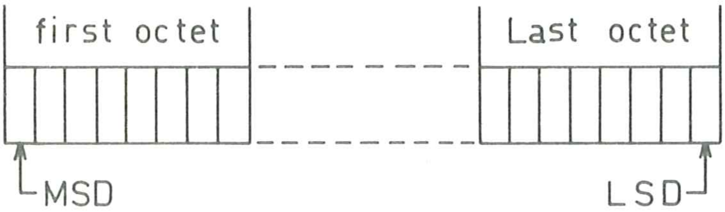
Example for the same 3-bit string fields as above:



UNSIGNED FIXED BINARY NUMBER:

Binary integer on the minimum number of octets corresponding to its declared range. The leftmost bit of the leftmost octet is the most-significant digit and the rightmost bit of the rightmost octet is the least-significant digit.

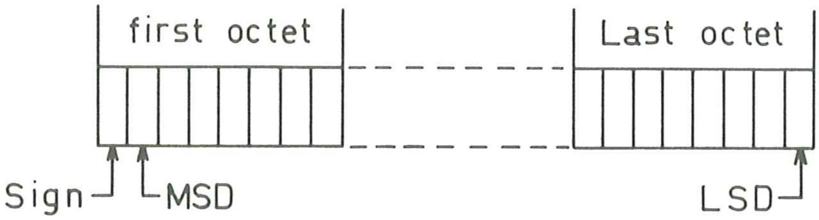
This can be illustrated as follows:



SIGNED FIXED BINARY NUMBER:

Binary integer on the minimum number of octets corresponding to its declared range. The second leftmost bit of the leftmost octet is the most-significant digit and the rightmost bit of the rightmost octet is the least-significant digit. Signed and in two's complement form. The sign is specified by the leftmost bit of the leftmost octet, which is equal to ONE if negative.

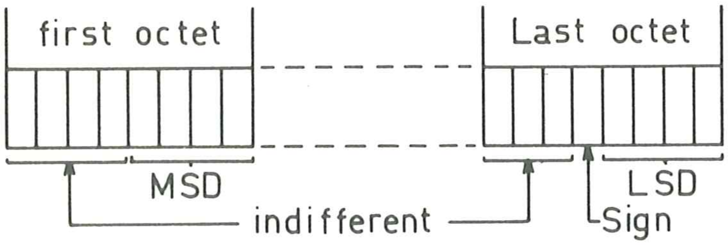
This can be illustrated as follows:



FIXED DECIMAL NUMBER (UNPACKED):

Series of decimal digits, each digit being right-justified in bits b5 to b8 of an octet. Digits are encoded in binary mode, from 0000 (decimal digit 0) to 1001 (decimal digit 9). Bits b1 to b4 of each octet are indifferently set, except for the rightmost octet of a signed decimal, which specifies the sign in bit b4 (0:positive, 1: negative). The leftmost octet contains the most-significant digit, the rightmost octet the least-significant digit.

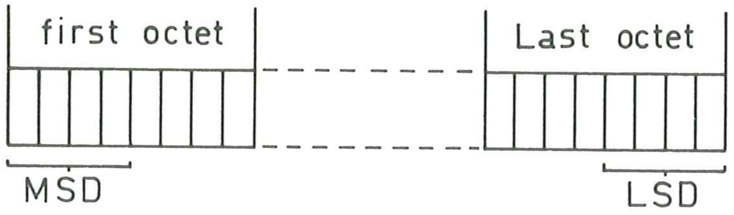
This can be illustrated as follows:



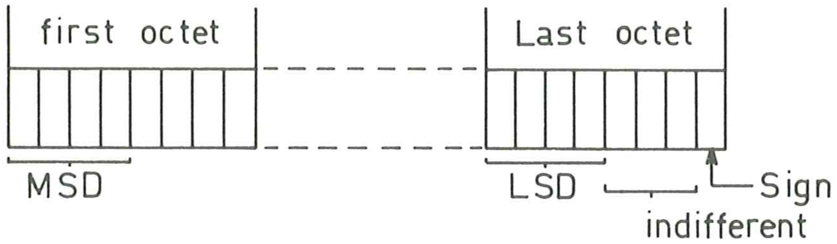
FIXED DECIMAL NUMBER (PACKED):

Same as unpacked, except each octet contains two digits: the most-significant of the two in b1 to b4, the least-significant in b5 to b8. When the range of the decimal number does not fill all available half octets, a ZERO is padded as the most-significant digit. When the decimal is signed, however, bits b5 to b8 of the rightmost octet do not contain a digit: bit b8 specifies the sign (0: positive, 1: negative), while b5 to b7 are indifferently set.

This can be illustrated as follows:



Unsigned fixed decimal number (packed)



Signed fixed decimal number (packed)

FLOATING POINT:

A floating point number x has seven characteristics:

- the number of significant digits of the fraction (or mantissa),
- the base b on which the digits of the fraction rely,
- the location of the radix point within the fraction,
- the value f of the fraction,
- the number m of significant digits of the exponent,
- the base on which the digits of the exponent rely,
- the value of the exponent.

For each floating-point number representation, the following is assumed:

1) The base b on which the digits of the fraction are built is equal to 16.

2) The value of the fraction is normalized, i.e.

$$1/b = 1/16 \leqslant |f| < 1 \text{ or } f = e = 0.$$

This implies that the most-significant digit of the fraction is not equal to zero if the number's value is not zero. The radix point is fixed at the left of the most-significant digit of the fraction.

3) The base on which the digits of the exponent rely is 2.

4) The representation of the floating point number x is built on the following formula:

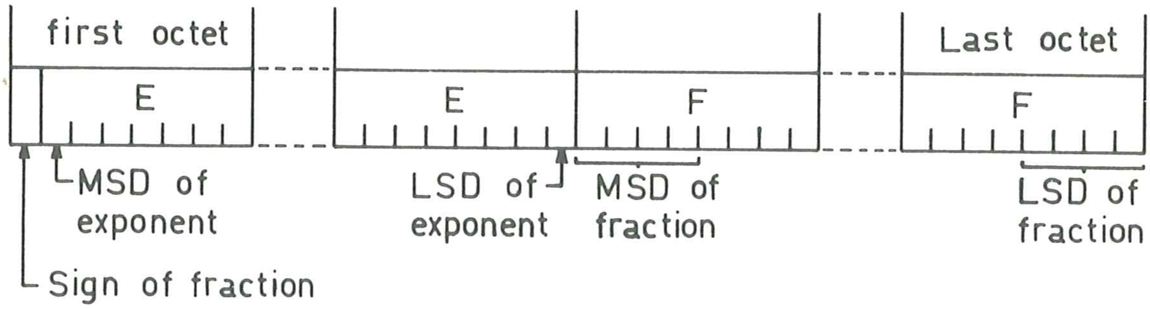
$$x = (-1)^s.F.16^{E-2^{m-1}}$$

where: $s = \text{sign } (f)$
$E = e + 2^{m-1}$
$F = \text{abs } (f)$
$m > 1$

5) The number is then represented as follows. The leftmost bit of the leftmost octet defines the sign of the fraction, which is equal to ONE if negative. To the right follows the exponent as an unsigned binary integer right-justified into the minimum number of octets, with ZERO as padding bits if necessary. According to rule (4), the encoded value E of the exponent complies to the formula:

$$E = e + 2^{m-1}$$

The fraction is represented by the given number of hexadecimal digits, with a minimum number of six digits (i.e. 3 octets). If the number of significant digits is odd, a ZERO is padded as the least significant digit. If a floating point is negative, its representation differs from the positive representation only at the sign location.



3.3.2.9 Compressed format

In this version, a single compression algorithm is defined: STD-1.

Data is divided into consecutive strings in an optimized TLV format.

T: String type (0: uncompressed, 1: compressed).

L: For T=0, length of the string in V,
for T=1, number of repetitions of the octet in V.

V: For T=0, L octets,

for T=1, 1 octet.

Type 0 strings decompress into the L octets in V; type 1 strings decompress into L times the octet in V.

The encoding is one octet for T and L: T in b1 and L in b2 to b8.

On the sending system, compression must take place after the data has been encoded according to the agreed syntax. On the receiving system, decompression is performed first.

3.4 SESSION SERVICE MAPPING

3.4.1 General

The data presentation protocol relies on the services offered by the Session Protocol (See ECMA-75).

3.4.2 Connection mapping

Each DPS connection uses one and only one session connection. The session connection is used by one and only one DPS connection.

NOTE 8

Future versions of the DPP may re-use the same session connection for multiple consecutive DPS connections.

NOTE 9

Multiplexing of multiple concurrent DPS connections onto a session connection is excluded in ISO 7498.

3.4.3 Session connection establishment

The establishment of a session connection between the two DPS entities is necessary before any DPP activity. The P-CONNECT service is mapped into the S-CONNECT primitives with the PP-CONNECT-RQ message forming the whole content of the ss-user-data parameter of the S-CONNECT request and indication primitives, and the PP-CONNECT-RP message forming the whole content of the ss-user-data parameter of the S-CONNECT response and confirmation primitives. The following parameters of CNQ and CNR are not mapped into ss-user-data but are mapped into other parameters of S-CONNECT primitives as in ECMA-75:

- user-caller into "initiator-address",
- user-called into "acceptor address",
- session-subset into "subset choice",
- session-parameters into "subset parameters" as relevant.

The result parameter of the S-CONNECT is set to "reject" if the diagnostic severity in PP-CONNECT-RP is "failure".

3.4.4 Session connection termination

The session connection is terminated in one of three ways:

- normal termination request (PP-RELEASE-RQ and -RP messages);
- forced termination request, resulting either from a P-DISCONNECT request primitive initiated by a DPS-user or a P-ABORT service element initiated by a DPS-entity; either case results in a PP-DISCONNECT-RQ message;
- spontaneous disconnection by the session service.

The first case is mapped onto the S-RELEASE service. The PP-CONNECT-RQ message is mapped into the ss-user-data parameter of the S-RELEASE request and indication primitives, and the PP-RELEASE-RP message is mapped into the ss-user-data parameter of the S-RELEASE response and confirmation primitives. The result parameter of S-RELEASE is set to "reject" if the diagnostic severity in PP-RELEASE-RP is "failure".

The second case is mapped onto the S-DISCONNECT service. The parameter of the PP-DISCONNECT-RQ forms the whole content of the ss-user-data parameter of S-DISCONNECT (See Note 10).

The third case is treated as a session service failure and signalled by the S-ABORT service. It is signalled to the DPS-users by local methods outside the scope of this Standard.

NOTE 10

The ss-user-data parameter of S-DISCONNECT is limited to three octets. The limit is expected to be removed in a future version.

3.4.5 Data exchange

The following DPP messages are mapped as session service data units of the S-DATA service:

PP-PERFORM-NEGOTIATION-RQ
PP-PERFORM-NEGOTIATION-RP
PP- DATA-RQ

3.4.6 Dialogue control facilities

The DPS services for dialogue control are directly mapped onto the equivalent session services, as follows:

P-SYNC	=	S-SYNC
P-END-DU	=	S-END-DU
P-RESYNC	=	S-RESYNC
P-TOKENS-GIVE	=	S-TOKENS-GIVE
P-PLEASE	=	S-PLEASE

If the DPS user has requested the use of a session data token, he is entirely responsible for the correct assignment of this data token before issuing a P-PERFORM-NEGOTIATION request or a P-DATA request. The DPS entity will spontaneously move the data token (using S-TOKENS-GIVE) only after sending a PP-PERFORM-NEGOTIATION-RQ or -RP message (in order that the acceptor of the negotiation be able to respond and that the data token be left unchanged for the user after the negotiation).

3.5 PROTOCOL SUBSETS

3.5.1 General

For each of the service subsets defined in 2.3, a corresponding protocol subset is defined in this clause. Dialogue facility messages apply to both subsets. Their availability depends only on the selected session subset.

3.5.2 Subset DP-A

The protocol messages included in this subset are:

PP-CONNECT-RQ and -RP
PP-RELEASE-RQ and -RP
PP-DISCONNECT-RQ
PP-DATA-RQ

3.5.3 Subset DP-B

The protocol messages included in this subset are the same as in subset DP-A, with the addition of:

PP-PERFORM-NEGOTIATION-RQ and -RP.

4 Conformance

4.1 CONFORMANCE REQUIREMENTS

4.1.1 General

This clause defines the conformance requirements for the Data Presentation Protocol defined in Section III of this Standard.

There is no conformance requirement for the abstract data presentation service defined in Section II of this Standard.

Only the externally visible and externally testable criteria are defined.

4.1.2 Equipment

The conformance requirement is for equipment which consists of hardware and/or software and has the purpose of conforming with this Standard. The equipment may also have other purposes.

4.1.3 Peer equipment

Any execution of the protocol necessarily involves a peer equipment with which the subject equipment communicates. For purposes of verifying conformance, it is assumed that this other peer equipment:

- is operating in conformance with this Standard,
- may be capable of controlled deviation, in that it may be the source of deliberate protocol errors for the purpose of testing.

This conformance requirement does not distinguish any differences of conformance status between the two equipments (i.e. the notion of a "reference equipment" with a "definitive implementation" is not used).

4.1.4 Protocol subsets

The equipment shall implement one or more of the protocol subsets defined in 3.5 and shall nominate which of these subsets the equipment is intended to conform with. It shall also nominate the session subsets with which the implementation is intended to work.

4.1.5 Additional data presentation protocol

In addition to the subsets nominated as in 4.1.4, the equipment may also implement other data presentation protocol, including different subsets of the protocol defined in this Standard.

Such additional provisions are themselves not in conformance with this Standard, but do not prejudice conformance with this Standard provided that they are separate and do not prevent use of the subsets nominated in 4.1.4.

4.1.6 Requirements

For each subset nominated as in 4.1.4 above, the equipment shall support establishment, use and termination of a DPS connection by execution of the protocol according to the following criteria. The subject equipment:

- shall accept correct sequences of DPP messages received from peer equipment, and respond with correct DPP message sequences, for the defined states of a DPS connection,
- shall respond correctly to all incorrect sequences of DPP messages received for a defined state of a DPS connection,
- shall accept all correct sequences of the session mapping,
- shall only issue correct sequences of the session mapping.

The terms "correct sequences" and "incorrect sequences" refer to the protocol defined in Section III and Appendix D.

Appendices

APPENDIX A

BRIEF DESCRIPTION OF THE REFERENCE MODEL OF OPEN SYSTEMS INTERCONNECTION

A.1 Scope

This Appendix is not part of the Standard. It is a copy of ISO/TC97/SC16 N 575.

This Appendix provides a brief description of the Reference Model of Open Systems Interconnection.

A.2 General Description

A.2.1 Introduction

The Reference Model of Open Systems Interconnection provides a common basis for the co-ordination of the development of new standards for the interconnection of systems and also allows existing standards to be placed within a common framework. The model is concerned with systems comprising terminals, computers and associated devices and the means for transferring information between these systems.

A.2.2 Overall perspective

The model does not imply any particular systems implementation, technology or means of interconnection, but rather refers to the mutual recognition and support of the standardized information exchange procedures.

A.2.3 The Open Systems Interconnection environment

Open Systems Interconnection is not only concerned with the transfer of information between systems (i.e. with communication), but also with the capability of these systems to interwork to achieve a common (distributed) task. The objective of Open Systems Interconnection is to define a set of standards which allow interconnected systems to co-operate.

The Reference Model of Open Systems Interconnection recognizes three basic constituents (See Fig. A1):

a) application processes within an OSI environment,

b) connections which permit information exchange,

c) the systems themselves.

NOTE A.1

The application processes may be manual, computer or physical processes.

A.2.4 Management Aspects

Within the Open Systems Interconnection architecture there is a need to recognize the special problems of initiating, terminating, monitoring on-going activities and assisting in their harmonious operations as well as handling abnormal conditions. These have been collectively considered as the management aspects of the Open Systems Interconnection architecture. These concepts are essential to the operation of the interconnected open systems and therefore are included in the comprehensivedescription of the Reference Model.

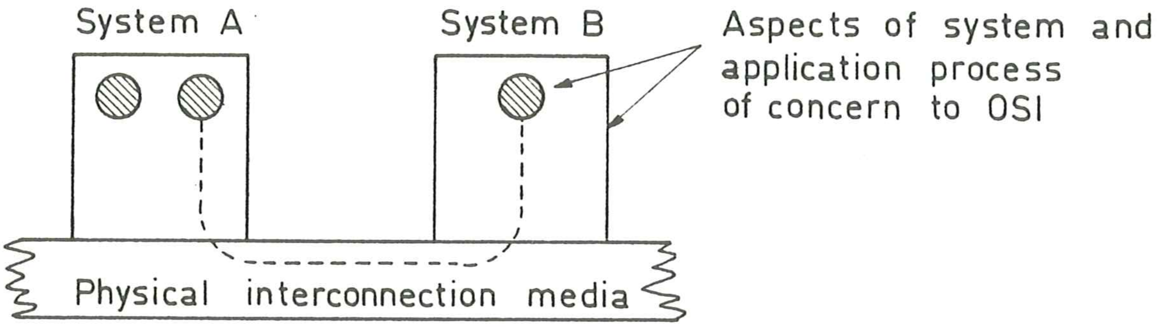

System A
System B
Aspects of system and application process of concern to OSI
Physical interconnection media

Fig. A.1 - General schematic diagram illustrating the basic elements of Open Systems Interconnection

A.2.5 Concepts of a Layered Architecture

The open systems architecture is structured in Layers. Each system is composed of an ordered set of sub-systems represented for convenience by Layers in a vertical sequence. Adjacent subsystems communicate through their common interface.

A Layer consists of all subsystems with the same rank. The operation of a layer is the sum of the co-operation between entities in that Layer. It is governed by a set of protocols specific to that Layer.

The services of a Layer are provided to the next higher Layer, using the functions performed within the Layer and the services available from the next lower Layer.

An entity in a Layer may provide services to one or more entities in the next higher Layer and use the services of one or more entities in the next lower Layer.

A.3 THE LAYERED MODEL

The seven-Layer Reference Model is illustrated in Fig. A 2.

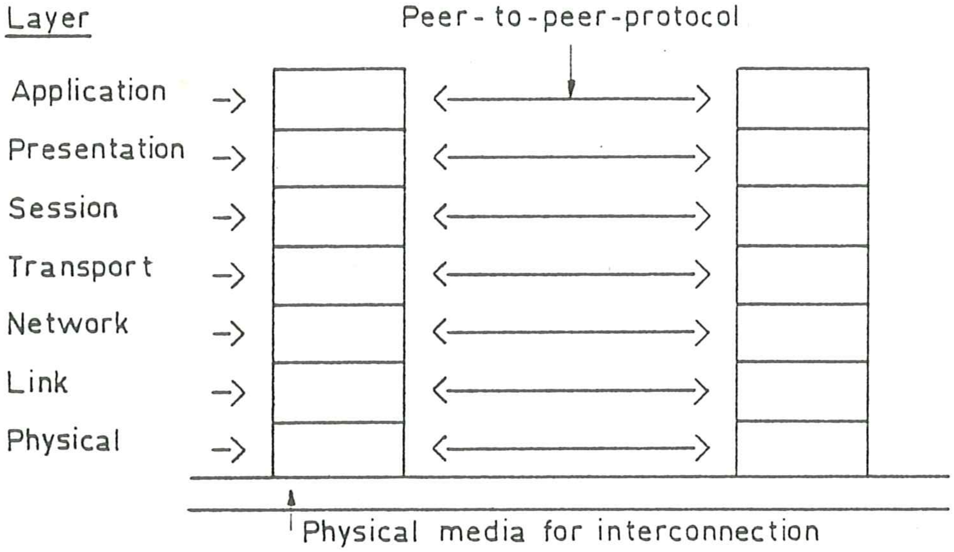


Fig. A.2 - The seven-layer Reference Model and peer-to-peer protocol

A.3.1 The Application Layer

As the highest Layer in the Reference Model of Open Systems Interconnection, the Application Layer provides services to the users of the OSI environment, not to a next higher layer.

The purpose of the Application Layer is to serve as the window between communicating users of the OSI environment through which all exchange of meaningful (to the users) information occurs.

The user is represented by the application-entity to its peer.

All user specifiable parameters of each communications instance are made known to the OSI environment (and, thus to the mechanisms implementing the OSI environment) via the application Layer.

A.3.2 The Presentation Layer

The purpose of the Presentation Layer is to represent information to communicating application-entities in a way that preserves meaning while resolving syntax differences.

The nature of the boundary between the Application Layer and the Presentation Layer is different from the nature of other Layer boundaries in the architecture.

The following priciples are adopted to define a boundary between the Application Layer and the Presentation Layer.

a) internal attributes of the virtual resource and its manipulation functions exist in the Presentation Layer;

b) external attributes of the virtual resource and its manipulation functions exist in the Application Layer;

c) the functions to use the services of the Session Layer effectively exist in the Presentation Layer;

d) the functions to use services of the Presentation Layer effectively exist in the Application Layer.

A.3.3 The Session Layer

The purpose of the Session Layer is to provide the means necessary for cooperating presentation-entities to organize and synchronize their dialogue and manage their data exchange. To do this, the Session Layer provides services to establish a session-connection between two presentation entities, and to support their orderly data exchange interactions.

To implement the transfer of data between the presentation-entities, the session-connection is mapped onto and uses a transport-connection.

A.3.4 The Transport Layer

The Transport Layer exists to provide the transport-service in association with the underlying services provided by the supporting layers.

The transport-service provides transparent transfer of data between session entities. Transport Layer relieves the transport users from any concern with the detailed way in which reliable and cost effective transfer of data is achieved.

The Transport Layer is required to optimize the use of the available communication resources to provide the performance required by each communicating transport user at minimum cost. This optimization will be achieved within the constraints imposed by considering the global demands of all concurrent transport users and the overall limit of resources available to the Transport Layer. Since the network service provides network connections from any transport entity to any other, all protocols defined in the Transport Layer will have end-to-end significance, where the ends are defined as the correspondent transport-entities.

The transport functions invoked in the Transport Layer to provide requested service quality will depend on the quality of the network service. The quality of the network service will depend on the way the network service is achieved.

A.3.5 The Network Layer

The Network Layer provides the means to establish maintain and terminate network connections between systems containing communicating application-entities and the functional and procedural means to exchange network service data units between two transport entities over network connections.

A.3.6 The Link Layer

The purpose of the Link Layer is to provide the functional and procedural means to activate, maintain and deactivate one or more data link connection among network entities.

The objective of this Layer is to detect and possibly correct errors which may occur in the Physical Layer. In addition, the Link Layer conveys to the Network Layer the capability to request assembly of data circuits within the Physical Layer (i.e. the capability of performing control of circuit switching).

A.3.7 The Physical Layer

The Physical Layer provides mechanical, electrical, functional and procedural characteristics to activate, maintain and deactivate physical connections for bit transmission between data link entities possibly through intermediate systems, each relaying bit transmission within the Physical Layer.

APPENDIX B

INDEX AND GLOSSARY OF TERMS

B.1 General

This Appendix is part of the Standard.

The terminology used in this Standard consists of:

- Terminology defined in the Reference Model for Open Systems Interconnection, which is defined in ISO 7498.
- Terminology for concepts of Presentation Layer in general and for Data Presentation Service and Protocol, in particular which is defined in this Appendix, See B.2.
- Notational terminology, which is defined in Appendix C.

B.2 Definitions

Compression: A reversible transformation of data in order to reduce its size without any loss of information. The reverse operation is called Decompression.

(Data) Field: The smallest unit of data of semantical significance. A given field occurence has a single value assigned.

Data Image Definition: An abstract description of DPS-user data. It is composed of an ordered list of field descriptions, where each field is characterized by its abstract data type and its size or range constraints.

Data Presentation Service (DPS): A service which provides an independence from specific local system data representations.

DPS Entity: An addressable entity to provide data presentation service.

Data Syntax: A defined representation of a given data type.

Data Type: An abstract characterisation of the contents of a given data field, independent of the syntax used for its representation.

Formatted Data: User data subject to transformation by the Data Presentation Service. It is composed of one or more fields.

Message: A unit of information exchange between two DPS entities. This unit is represented either by a presentation protocol data unit (ppdu) and a pair of session service primitives or by a pair of session service primitives only.

Transfer Environment (TE): An agreed upon set of data types and associated syntaxes (including possibly a compression method) used at a given time on a DPS connection.

Transparent Data: User data not subject to transformation by the Data Presentation Service.

APPENDIX C

NOTATION

C.1 Introduction and scope

This Appendix is part of the standard.

It is a reproduction of notation defined in the ECMA register of common techniques for OSI standards, particularized by substitution of the acronym "DPS" into the terms defined.

C.2 Definitions

This terminology is for the notation defined in this Appendix.

(x)-service: A conceptual unit of the data presentation service, of which (x) is its particular name.

(Service) primitive: A discrete component of an (x)-service.

(x)-Request primitive: A type of primitive by means of which a DPS user causes an occurence of the (x)-service.

(x)-Indication primitive: A type of primitive by means of which a DPS user is informed of an occurence of the (x)-service.

(x)-Response primitive: A type of primitive by means of which a DPS user replies to an occurence of an (x)-indication primitive.

(x)-Confirmation primitive: A type of primitive by means of which a DPS user is informed of an occurence of an (x)-response primitive.

Service structure: The series of one or more primitives of which an (x)-service wholly consists.

Service structure type 1: A service structure with a request primitive and an indication primitive (non confirmed).

Service structure type 2: A service structure with a request primitive, an indication primitive, a response primitive and a confirmation primitive (confirmed).

Service structure type 3: A service structure with two indication primitives (indication only).

Event: The occurence of a primitive.

Initiator: The DPS user who issues the request primitive of the (x)-service concerned.

Acceptor: The DPS user who receives the indication primitive of the (x)-service concerned.

C.3 Service model

The data presentation service is modelled as an abstract service to which DPS users gain access at DPS-access-points. All interactions are between two DPS users located at separate DPS-access-points. A single DPS connection is modelled.

C.4 Primitives

The data presentation service is defined by means of service primitives.

Primitives are conceptual and are not intended to be directly related to data presentation protocol elements or to the units of interaction across a procedural interface in an implementation. The descriptive technique is independent of such variable details.

Primitives which relate only to local conventions between a DPS user and an implementation are not defined.

The subdivision of the data presentation service into the particular primitives chosen is arbitrary, in that the same data presentation service could be described by other logically equivalent primitives. There is no notion that a primitive is "elementary".

A primitive occurs at one service access point (not both). It usually has parameters, containing values related to its occurence.

The occurence of a primitive is a logically instantaneous and indivisible event. The primitive occurs at a logically separate instant, which cannot be interrupted by the occurence of another primitive. It occurs either completely or not at all.

There are four types of primitives in this Standard (See C.2):

a) request primitive
b) indication primitive
c) response primitive
d) confirmation primitive

The primitives are given names prefixed by "P" (presentation) to distinguish them from primitives of adjacent layers. The names of the primitives are written in upper case, e.g. P-DATA.

C.5 Service structure

Each service consists of one or more primitives and affects both service access points. There are three different combinations of primitives. These combinations are referred to as service structures. The three service structure types used in this standard are defined in C.2.

Unlike the occurence of its constituent primitives, the occurence of a service structure is not logically instantaneous and indivisible. The intervals between its constituent primitives may be non-disruptively interspersed with the primitives of other service structures, subject to restrictions particular to the service concerned. Service structures may also be disrupted by the occurence of certain other primitives (See C.6).

C.6 Effects of services

The effects of a service are referred to as being sequentially transmitted if its successive primitives at one service access point result in the same sequence of corresponding primitives at the other service access point (unless disrupted, see below).

The effects of a service are referred to as being expedited if its indication or confirmation primitives may arrive at the other service access point before those of a previous occurence of a service.

The effects of a service are referred to as being disruptive if it may destroy, and therefore prevent the occurence of, indication and confirmation primitives corresponding to previous request or response primitives. The effects of disruptive services are expedited, unless stated otherwise.

The effects of a primitive are referred to as being non-disruptive if they do not have the above disruptive effects. Non-disruptive effects may include effects relating to or delaying other events without destroying them.

Unless otherwise specified, the effects of a service are sequentially transmitted and non-disruptive.

C.7 Parameter notation

For each service structure, the parameters are defined by a table, followed by a list of parameter descriptions. The column headings in the parameter tables indicate the primitive types: REQ for Request, IND for Indication, RESP for Response, CONF for Confirmation.

The values in the columns of the parameter tables obey the following conventions:

D (down):	value supplied by the DPS user in the primitive
U (up):	value supplied by the DPS in the primitive
B (both):	value supplied either by the user or by the DPS in the primitive

x: parameter not used in the primitive.

The detailed description of a parameter includes its purpose, the rule for setting its value, the default value and the legal values. All parameters which are supplied by the user (D) both in request and response primitives are negotiated. The others are not negotiated.

Unless otherwise stated, the parameter value in the indication is the same as that in the request, and the parameter value in the confirmation is the same as that in the response.

Parameter values are only defined to a level which distinguishes meaning, but generally without defining their absolute value or encoding. These details are outside the scope of the standard, being local conventions between the DPS user and an implementation.

APPENDIX D

FORMAL DESCRIPTION

D.1 Introduction

This Appendix is part of the Standard.

In section 3 of this Standard, the data presentation protocol interactions between two DPS entities are described. That description references states, events and actions which in this appendix are consolidated into a formal description of the protocol, as a Finite State Machine (FSM).

NOTE D.1

There is no formal description of the session mapping.

D.2 Elements used in the formal description

Table D.1 lists the states which are used in the formal description. For each entry there is a state code and a brief description.

Table D.2 lists the events which are used in the formal description. For each entry there is an event code and a brief description.

Table D.3 lists the message acronyms which are used in the formal description to identify messages sent.

Table D.4 lists the conditions which are used in the formal description. For each entry there is a condition code and a brief description.

The way in which these various elements are used is defined in D.3.

Table D.1 - FSM states

State-code	State-description	State tables
CF..	(Connection Facility states)	
CF01	Unconnected	D.5
CF02A	CN-pending (local request)	D.5
CF02B	CN-pending (remote request)	D.5
CF03A	RL-pending (local request)	D.5,6,7
CF03B	RL-pending (remote request)	D.5
PF..	(Presentation Facility states)	
PF01	Data	D.5,6,7,8
PF02A	PN-pending (local request)	D.5,6,7,8
PF02B	PN-pending (remote request)	D.6
DF..	(Dialogue Facility states)	
DF01A	END-DU-pending (local request)	D.6,7,8
DF01B	END-DU-pending (remote request)	D.7
DF02A	RESYNC-pending (local request)	D.7
DF02B	RESYNC-pending (remote request)	D.7

Table D.2 - FSM events

Event-code	Event description
XXQ	XXQ request message
XXR	XXR response message
XX-RG	Request primitive associated to XX
XX-IN	Indication primitive associated to XX
XX-RP	Response primitive associated to XX
XX-CF	Confirmation primitive associated to XX

NOTE D.2

Table D.3 gives the list of valid values for XX. An example is given below, where X = CN.

CNQ	*:*	*CNQ request message*
CNR	*:*	*CNR response message*
CN-RG:		*P-CONNECT request primitive*
CN-IN:		*P-CONNECT indication primitive*
CN-RP:		*P-CONNECT response primitive*
CN-CF:		*P-CONNECT confirmation primitive*

Table D.3 - List of protocol messages

Acronym	Message name	State tables
CN	Connect	D.5
RL	Release	D.5
DN	Disconnect	D.5
AB(*)	Abort	D.5
PN	Perform Negotiation	D.6
DT	Data	D.6
SYNC	Synchronize	D.7
PLEASE	Please tokens	D.8
TK-GIVE	Give tokens	D.8
END-DU	End dialogue unit	D.7
RESYNC	Resynchronize	D.7

(*) There are only two events related to the acronym AB:
AB: Session connection abort indication from layer 5
AB-IN: DPS connection abort indication

Table D.4 - Conditions

Condition	Meaning
+	Value of diagnostic severity is "success" or "success with warning"
-	Value of diagnostic severity is "failure"
nt	Holder of negotiate tokens is local entity
dt	Holder of data token is local entity
st	Holder of synchronize token is local entity
tt	Holder of terminate token is local entity
cw	Local entity is contention winner
CONDV:cond	This intersection is valid only if cond is true

NOTE D.2

nt (negotiate tokens) is an abbreviation for (dt and st and end-du-token). See ECMA-75 for the definition of elementary tokens.

NOTE D.3

xt and ^xt are true if the x token is not defined.

NOTE D.4

cw is used for collision of RESYNC requests. The criteria for cw to be true are defined in ECMA-75, subclause 3.2.5.2.

D.3 Formal description conventions

The formal description is specified in Tables D.5 to D.8.

The horizontal dimension of each table is the set of all the states. If for any given state there is no valid event, then the state does not appear in the table, i.e. no column.

The vertical dimension of each table is the set of all relevant FSM request events, incoming message events and FSM response events. For each event there is an entry, i.e. a row.

Each valid intersection contains:

- one or more conditions (where relevant);
- one or more actions (where relevant);
- the new state (always).

The conditions are those defined in table D.4.

The action consists of sending a message or issuing a local primitive event (indication or confirmation) or setting a condition.

The new state is the state which is entered after the specified actions are completed.

When a request message is dropped as a result of contention solving, no state change occurs and the indication primitive is not issued.

An intersection is invalid either if it contains a blank entry or if the condition indicated by CONV is not met. All invalid intersections between states and incoming message events are treated as protocol violations: the action is disconnect the DPS connection (DN message with "protocol violation" reason-code), the new state is CF01 (unconnected).

All invalid intersections between states and incoming primitive events are treated as local errors, in a way not specified in this Standard.

The following editorial conventions are used:

: = action with regard to state change

, = connects list of actions

(a&b)= logical 'and' of conditions a and b

^ = logical negation

D.4 Formal description tables

Because the complete state table for the data presentation protocol is too large to be edited on a single page, it has been segmented into smaller tables:

table D.5: connection facility

table D.6: presentation facility

table D.7: dialogue facility (synchronization)

table D.8: dialogue facility (token management)

Table D.5 - Connection Facility

	CF01 UNCONN	CF02A CN-PEND FROM LOCAL	CF02B CN-PEND FROM REMOTE	CF03A RL-PEND FROM LOCAL	CF03B RL-PEND FROM REMOTE	PF01 DATA	PF02A PN-PEND FROM LOCAL	DF01A END-DU-PEND FROM LOCAL
CN-RQ	: CF02A CNQ							
CNR		CN-CF +:PF01 -:CF01						
CNQ	CN-IN : CF02B							
CN-RP			+:PF01 -:CF01 CNR					
RL-RQ						CONDV: (tt&dt) :CF03A RLQ		
RLR				RL-CF +:CF01 -:PF01				
RLQ						CONDV: (^tt&^dt) RL-IN :CF03B	CONDV: (^tt&^dt) RL-IN :CF03B	CONDV: (^tt&^dt) RL-IN :CF03B
RL-RP					+:CF01 -:PF01 RLR			
DN-RQ (1)		:CF01 DNQ	:CF01 DNQ	:CF01 DNQ	:CF01 DNQ	:CF01 DNQ	:CF01 DNQ	:CF01 DNQ
DNQ (1)		DN-IN :CF01	DN-IN :CF01	DN-IN :CF01	DN-IN :CF01	DN-IN :CF01	DN-IN :CF01	DN-IN :CF01
AB (1)		AB-IN :CF01	AB-IN :CF01	AB-IN :CF01	AB-IN :CF01	AB-IN :CF01	AB-IN :CF01	AB-IN :CF01

(1) The effect of this event is the same for any other state not explicitly mentioned in this table.

Table D.6 - Presentation Facility

	CF03A RL-PEND FROM LOCAL	PF01 DATA	PF02A PN-PEND FROM LOCAL	PF02B PN-PEND FROM REMOTE	DF01A END-DU- PEND FROM LOCAL			
DT-RQ		CONDV:dt :PF01 DTQ						
DTQ	CONDV:^dt DT-IN :CF03A	CONDV:^dt DT-IN :PF01	CONDV:^dt DT-IN :PF02A		CONDV:^dt DT-IN :DF01A			
PN-RQ		CONDV:nt :PF02A PNQ						
PNR			PN-CF :PF01					
PNQ	CONDV:^nt :CF03A	CONDV:^nt PN-IN :PF02B						
PN-RP				:PF01 PNR				

Table D.7 - Dialogue Facility (Synchronization)

	CF03A RL-PEND FROM LOCAL	PF01 DATA	PF02A PN-PEND FROM LOCAL	DF01A END-DU-PEND FROM LOCAL	DF01B END-DU PEND FROM REMOTE	DF02A RESYNC-PEND FROM LOCAL	DF02B RESYNC-PEND FROM REMOTE	
SYNC-RQ		CONDV: (dt&st) :PF01 SYNCQ						
SYNCR	SYNC-CF :CF03A	SYNC-CF :PF01	SYNC-CF :PF02A	SYNC-CF :DF01A				
SYNCQ	CONDV: (^dt&^st) SYNC-IN :CF03A	CONDV: (^dt&^st) SYNC-IN :PF01						
SYNC-RP		:PF01 SYNCR						
RESYNC-RQ		:DF02A RESYNCQ					cw:DF02A RESYNCQ ^cw:DF02B	
RESYNCR						RESYNC-CF :PF01		
RESYNCQ	RESYNC-IN :DF02B	RESYNC-IN :DF02B	RESYNC-IN :DF02B	RESYNC-IN :DF02B		cw:DF02A ^cw:DF02B RESYNC-IN		
RESYNC-RP							:PF01 RESYNCR	
END-DU-RQ		CONDV:nt :DF01A END-DUQ						
END-DUR				END-DU-CF :PF01				
END-DUQ	:CF03A	CONDV:^nt END-DU-IN :DF01B						
END-DU-RP					:PF01 END-DUR			

Table D.8 - Dialogue Facility (Token Management)

	CF03A RL-PEND FROM LOCAL	PF01 DATA	PF02A PN-PEND FROM LOCAL	DF01A END-DU-PEND FROM LOCAL				
PLEASE-RQ		:PF01 PLEASEQ						
PLEASEQ	PLEASE-IN :CF03A	PLEASE-IN :PF01	PLEASE-IN :PF02A	PLEASE-IN :DF01A				
TK-GIVE-RQ		:PF01 TK-GIVEQ						
TK-GIVEQ	TK-GIVE-IN :CF03A	TK-GIVE-IN :PF01	TK-GIVE-IN :PF02A	TK-GIVE-IN :DF01A				

NOTE D.5

TK-GIVE-RQ valid only for the current owner of the specified token(s). TK-GIVEQ valid only if the specified token(s) are not currently owned by the receiver of the message.

APPENDIX E

USE OF THE SPECIAL EXTENSION MECHANISM

E.1 Scope

This Appendix is not part of the standard. It is provided as an aid to implementations of the standard.

The services and protocols defined in this document apply to fully heterogeneous networks. As such, they cover only general needs, since they cannot reasonably include features which are not widely supported in current file systems. However, in many cases, they will be used within less heterogeneous networks (or parts of networks) where the systems or the end users can agree to particular common conventions. For such cases, a mechanism is provided whereby special extensions can be unilaterally defined and implemented, according to particular conventions, and without degrading the openness to fully heterogeneous environments.

Particular conventions may serve two main purposes:

- addition of specific services or data models.
- simplification, by assuming implicit prenegotiation of a number of parameters (useful for small systems offering a single choice for each capability).

If particular conventions have been used for general needs not satisfied in an early version of the DPP, it is strongly advised to abandon them as soon as the corresponding needs are satisfied by a new version of the DPP.

E.2 Special extension mechanism

A pre-requisite is to recognize if the other entity supports the same conventions. This is achieved through the special-conventions parameter of the P-CONNECT service, allowing to negotiate any number of special conventions (by repetition of the parameter). The value of this parameter is an arbitrary string of 1 or 2 octets identifying a set of conventions.

When at least one set of special conventions has been retained the following is allowed in the other services:

- omission of mandatory parameters: their value is either pre-negotiated or supplied by special extensions.
- usage of private values in all parameters where such values are allowed.

- usage of the special-information parameter, where applicable, for conveying special extensions. If several special parameters are required for a given service, two possibilities are offered:

  . repeat the special-information parameter

  . use any encoding scheme allowing to group several distinct special parameters into a single occurence of the special-information parameter (e.g. the encoding scheme defined in 3.3.2.1).