ECMA
EUROPEAN COMPUTER MANUFACTURERS ASSOCIATION

STANDARD ECMA-87

GENERIC VIRTUAL TERMINAL

SERVICE AND PROTOCOL DESCRIPTION

March 1983

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-87

GENERIC VIRTUAL TERMINAL

SERVICE AND PROTOCOL DESCRIPTION

March 1983

BRIEF HISTORY

This Standard ECMA-87 has been developed by TC23 and adopted as an ECMA Standard at the General Assembly of Dec. 16, 1982.

This Standard ECMA-87 constitutes the present ECMA position as a technical contribution to ISO and CCITT to achieve world-wide standardization. However, ECMA does not yet consider this document to be a basis for product compliance as there is still international work going on.

Introduction

INTRODUCTION

This Standard ECMA-87 is one of a series of standards for Open Systems Interconnection.

Open Systems Interconnection standards are intended to facilitate homogeneous interconnection between heterogeneous information processing systems.

The standard is within the framework for the coordination of standards for Open Systems Interconnection which is defined by the ISO Reference Model of Open Systems Interconnection (OSI), ISO 7498.

It 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, the CCITT, and national standards 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 interests of rapid and effective standardization, the standard is orientated towards urgent and well understood needs. It is intended to be capable of modular extension to cover future developments in technology and needs.

The Virtual Terminal Service Within the Presentation Layer

The ISO Reference Model of Open Systems Interconnection (OSI) is now a well accepted basis for the development of standards for the purpose of systems interconnection. A frequent case of open working involves human partners communicating with application partners through a large variety of terminals. The Virtual Terminal Service (VTS) is a service of the Presentation Layer of the OSI Model, intended to disguise the existing incompatibilities between physical terminals offering similar functions. The VTS offers to the application program a logical view of real terminals, called Virtual Terminal, on which a logical information structure can be defined and on which corresponding standard functions can be performed. The VTS provides

for independence from physical devices in terms of the syntax of the interactions but does not attempt to control the semantics of the interactions, this being the function of the Application Layer services and protocols. The application-entity partners, of whatever nature, using the VTS are presentation service users (p-users), and this term is used throughout. In Open Systems Interconnection the two p-users are considered to be attached to the open systems environment by local systems functions which are responsible for the mapping of the interconnection functions (at all layers of the ISO Model) to the particular form of user interfaces provided in the local system; these interfaces may be in the form of conventional "terminals" or "programming" interfaces, etc. Due to this symmetry in the OSI concept the VTS is also defined in a symmetrical way so that no distinction is made between users of the above general types.

The ECMA VTS itself operates in p-layer in the framework of the Generic Presentation Services (GPS) described in Standard ECMA-86, Generic Presentation Services - Description and Protocol Definition, as a specific presentation service as defined therein. The corresponding Virtual Terminal Protocol (VTP) (actually one of a number of classes of VTP) operates as a specific presentation protocol operating within the framework of the Generic Presentation Protocol (GPP) as defined in Standard ECMA-86.

The purpose of this Standard, ECMA-87, is to introduce and describe a Generic Model of the Virtual Terminal, with reference to which the details of the Virtual Terminal Service can be described and hence the Virtual Terminal Protocol (VTP) standards can be defined. The interpretation of the protocol will be described in terms of operations on the various components of the model. The abstract model and the conceptual service interface are introduced only as an aid to the description of the functions required from the service and hence the functions for which a protocol must be designed to cater. The Service Description will be the basis for the design of application layer functions.

While the fundamental facilities of a VTS are the same for supporting all types of real devices (e.g. establish connection, exchange data and commands, relay status information, terminate connection, etc.), some of the associated service elements will vary widely due to the wide range of real devices required to be supported (e.g. character based as opposed to graphics, line and page orientated as opposed to forms orientated, binary, etc.) and also the wide range of applications requiring to use, or be used by, these devices. As a result of this diverse requirement the concept of Class of Virtual Terminal Service is introduced. This allows a number of distinct classes of service to be identified, each of which exhibits a set of logical functions broadly reflecting various categories of usage of real terminals (as far as functionality is concerned). Each of these classes will reflect substantially different functions with minor variations being

handled by the selection of options within a particular class.

The introduction of distinct classes within the VTS does not preclude the possibility of handling real terminals which exhibit functions characteristic of more than one class as this is implementation dependent and could for example be handled by allowing more than one class of service to be mapped onto the same real terminal (at different points in time) depending on the service required at a particular point in time. It can therefore be stated that although the Virtual Terminal protocols will be standardized (and as a result the basic functional features of the VTS) any local VTS service interface and mapping of the Virtual Terminal onto a real device(s) is by nature implementation dependent and beyond the scope of both the VTS and the model introduced here.

For each class of service so identified a separate Virtual Terminal Protocol Standard will be defined with reference to the Generic Model described in this Standard, which is applicable to all classes. All such Class standards will be compatible with the principles of the GPP.

TABLE OF CONTENTS

Table of Contents (cont'd)

1 General

1. GENERAL

1.1 Scope

This Standard ECMA-87

a) defines the terminology, concepts, descriptive model and notation for generic aspects of the Virtual Terminal Service (VTS) in the Presentation Layer of the Reference Model of Open Systems Interconnection, (ISO 7498), including the criteria for classification of virtual terminal services,

b) defines in abstract form the interactions between two presentation-service-users (p-users) via the Virtual Terminal Service of Open Systems Interconnection,

c) gives in general terms an overview of the protocol between two presentation entities (p-entities) providing the presentation service (p-service), in particular VTS,

d) gives in general terms an overview of class-independent aspects of the usage of services of the Session Layer of OSI.

Appropriate relationships are established with the Generic Presentation Services GPS and Protocols GPP.

This Standard ECMA-87 does not define any other interactions between a presentation service user and the presentation service.

This Standard ECMA-87 is not an implementation specification for information processing systems.

This Standard ECMA-87 contains no explicit conformance statements but is referred to by individual VTS/P Class Standards and certain parts thereby acquire an implicit conformance requirement.

1.2 Field of Application

This Standard ECMA-87 is provided in order to be referenced by other standards, and in particular by:

a) Standards specifying detailed presentation protocol(s) for the interactions defined,

b) Standards for the Application Layer which use the interactions defined.

1.3 References

ECMA-75 : Session Protocol

ECMA-86 : Generic Data Presentation - Services Description and Protocol Definition

ECMA-88 : Basic Class Virtual Terminal - Service Description and Protocol Definition

DIS 7498 : Reference Model of Open Systems Interconnection

NOTE 1

In this Standard, Standard ECMA-86 is commonly given the acronyms GPS for service aspects and GPP for protocol aspects.

2 Service

2. SERVICE

2.1 The Generic Model of Virtual Terminal Services

In order to describe Virtual Terminal Services in a consistent way an abstract Generic Model of the Virtual Terminal is used. After an introduction to the components of the model there follows a description of the general concepts and mechanisms which will allow particularization of the components of the model and access to, interpretation and manipulation of, the information content of the various components. It is important to note that the model described here is intended to be general enough to apply to any class of service and exact definitions and mechanisms for operation on the model will be defined in the different VT standards for the various Classes of Virtual Terminal.

The Model of VTS described in this Standard in generic terms and in detail in specific class standards is to be viewed as operating within the framework of the Generic Presentation Services (GPS) described in Standard ECMA-86. GPS establishes the concept of Presentation Environment (PE) and Transfer Enclosure and all "useful" activity of a VTS occurs within this context of GPS. The other aspects of GPS such as Establishment, Termination, Negotiation Facilities, and Enclosure Control are made use of as required to set up the context for useful operation. Multiple Presentation Environment operation is available. Individual classes and subsets of classes will not necessarily make use of all these GPP facilities.

Primarily the model consists of a (single) Conceptual Communication Area (CCA) which contains all the necessary information to allow the communicating p-users (application(s) and/or terminal(s)) to derive a consistent view of the Virtual Device(s) which comprise the Virtual Terminal. The CCA is a conceptual repository for all information being shared by the p-users. The CCA will not have a physical existence but each p-user will normally have its own realisation of it.

The CCA is the particular form of presentation environment (PE) used by VTS. This Generic Standard is concerned only with the generic nature of the PE for VTS and the operations which can be performed on it. Particular Class Standards will extend or add detail to the generic statements as necessary.

Figure 1 illustrates the VTS model, and is followed by notes. It is a fully abstract model which conceptualizes the VTS into a common (in concept, though probably distributed in practice) conceptual service interface accessed by each

p-user. The real requirements of the users should always be first considered using this conceptualization. If they cannot be fitted into this framework then VTS is probably not the appropriate service for these users.

As shown in Figure 1 the CCA contains the following component information structures:

- Data Structure Definition (DSD)
- Conceptual Data Store (CDS)
- Access Control (AC)
- Control, Signalling and Status Store (CSS)

Using the above information structures and their contents stored in the CCA each p-user is able to derive a visualization of the Virtual Terminal. Such a visualization implies the following:

- The p-user is able to visualize an image of the data content of the CDS; this image is referred to as the Conceptual Image, which may be routed to any of the virtual devices,
- the p-user is able to modify in an orderly manner, the content of the CDS, including the attribute information, if present,
- the p-users are able to control, to signal to, and to obtain status information from, the Virtual Devices of the Virtual Terminal by means of conceptual mechanisms, the data associated with such control signalling and status flows being held in appropriately defined areas in the CSS.

Figure 1 illustrates this model. Some further explanation is given below.

Figure 1 shows the two p-users sharing a common conceptual communication area containing AC, DSD, CDS, CSS, and the common conceptual service interface to this. The precise nature of each of the components is decided when the service is established, by the specified class of service and negotiated parameters within the class. Each p-user writes information into the CCA by means of conceptual service interface requests and at the conceptual level the data is "made accessible" - it does not matter at this conceptual level whether it is "delivered" or is merely "made readable". The detailed definition in this Standard states that it is in fact "delivered" to the peer p-user by use of the access control facilities (delivery control and give-token).

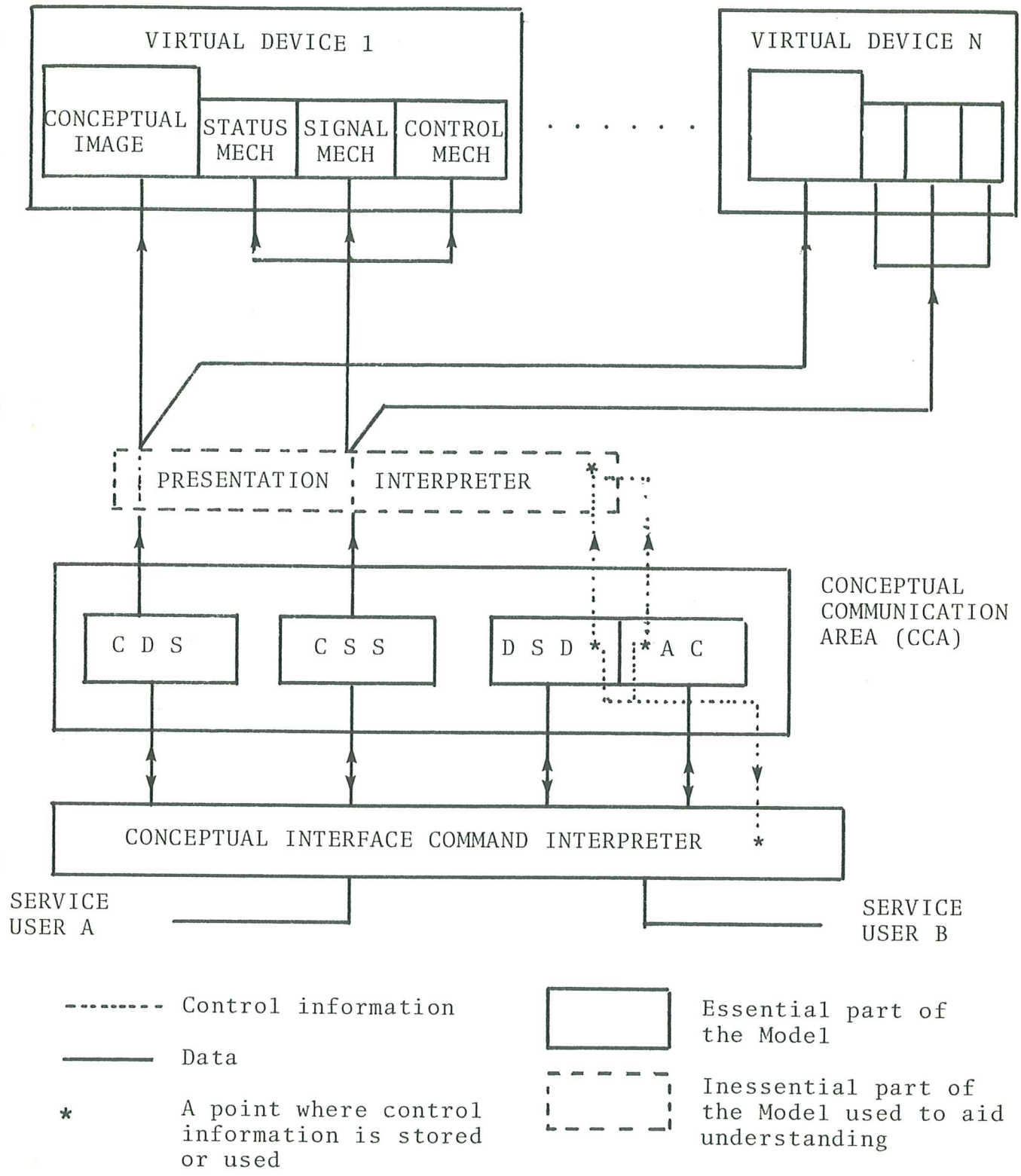


Figure 1. The Model View (Conceptualization of Service)

While all the information in the CCA is shared by the p-users (there is no point in putting information private to one or the other of them in the CCA) some kind of access control discipline is required. The diagram indicates this by showing the routing of information to, and from, the p-users through an interpreter which uses information residing in the AC. Whether a p-user currently has the right to write (update) or read (access) a particular piece of information at a particular time is assumed to be controlled in the conceptual model by the access control information in AC. Some of this information by its very nature is under the control of the VTS itself so that in effect neither p-user has the (direct) right to modify it. However, it is conceptually viewed as being in the AC component of the CCA since both p-users need to see it.

Figure 1 also shows the conceptual visualization of the information in the CDS as a conceptual image on one or more Virtual Device(s), interpreted in accordance with any "realization" parameters in the DSD. These may also determine the conceptual relation between the CSS data areas and status control and signalling mechanisms conceptually associated with one or more of the virtual devices. Note particularly that it is the service users which enter information into the CSS, the virtual devices are not sources of information flowing into the CCA.

Note 2

The postulation of virtual device and conceptual image are convenient for clarity of description; in the particular case of two applications communicating using VTS there would not be any physical device.

A diagram and explanatory notes on the implementation view are included in Section III.

2.1.1 Virtual Terminal Classification

The Virtual Terminal Model allows its components to be defined, structured and accessed in different ways according to the functions required. A single Virtual Terminal Protocol standard to cover the total range of possibilities would be extremely unwieldy, difficult to comprehend and very expensive to implement and require frequent updating of this Standard for new requirements. At the other extreme, separate protocols for each type of real device and mode of usage would require a very large number of different implementations and is equally unattractive (it is the situation existing currently which the VTS/VTP concept is intended to improve). The concept of Class of Usage of VTS is introduced with the aim of striking a balance between these extremes, enabling practical implementations for current terminal types and usages, while offering a framework for continued development of more advanced features.

It is intended that a new class will be introduced only where there is some fundamental difference in the service offered. Each class is expected to offer a wide range of parameterization of the detailed characteristics of the functions available within its scope. As an example of this the wide range of capabilities offered by the 1,2, or 3 dimensional CDS in Basic Class may be examined; some possibilities are given in Appendix E of Standard ECMA-88.

Two principle criteria have been established for the classification of Virtual Terminal usage, as follows:

(1) the type(s) of Atomic Objects to be held in the Cells of the CDS; e.g. point, character, binary, vector graphic picture element, facsimile, etc.,

(2) the structuring of the organization of, and the relationship between, the cells of the CDS; e.g. independent, implied ordered array, areas of contiguous cells, etc.

The current set of ECMA Classes of VTS is based solely on these criteria as shown in the table below. Extensions to this table are likely.

ATOMIC OBJECT	NO STUCTURING	HIERARCHICAL GROUPING
CHARACTER	BASIC CLASS	FORM CLASS
BIT	IMAGE CLASS	-
GRAPHIC ELEMENT	-	GRAPHIC CLASS
MIXTURE OF ABOVE	-	MIXED-MODE CLASS

It is important to note that these criteria distinguish the functional capabilities of one class of service from another. An application will decide what type of service it requires, regardless of the physical terminal capabilities. A given physical terminal may well be able to support a variety of classes of service. For example a sophisticated vector graphics terminal can support trivial alphanumeric applications using the simplest (Basic Class) protocol, whereas the converse is frequently not true - a character-orientated terminal is unlikely to be able to support a graphics application unless it has comprehensive mosaic characters or user-definable raster characters.

For character-orientated applications an approriate protocol may be either BASIC Class where no structuring is required - the CDS is an ordered one, two, or three dimensional array of character cells - , or FORM Class where a hierarchical grouping of character cells into sub-areas and areas (sub-fields and fields, sub-maps and maps, the hierarchical concept is important, the terminology is not) is required. In general an application should demand the simplest mode of usage with which it can manage, a terminal should provide the maximum features possible with its real devices; in this way the compatible connections are maximized in the Open Systems environment. (A negotiation attempt to obtain more than the absolute minimum requirement is of course possible.)

For graphical applications hierarchical structuring appears to be essential (basic elements are hierarchically grouped into objects, sub-pictures and pictures in all successful graphic support packages), and there is therefore, no requirement for an equivalent of the Basic Class structure, similarly for Mixed Mode where graphics elements are an inherent part of expected applications.

A case which has not yet been studied in detail and which may give rise to a further Class is that of TEXT manipulation. Although Basic and Forms classes can provide capabilities adequate for many cases it is possible that provision of a special class with some specilized facilities may be seen as desirable. However, the specific criterion which would justify this has not yet been identified.

2.2 Service Overview

This clause describes the generic form of the Conceptual Communication Area used as the basis of the Model of VTS, which is divided into a number of components as described in the following sub-clauses. Clause 2.3 gives further detail of the operations which can be performed on these components.

For convenience the acronym GVT is used in this Standard and in individual class standards to refer to generic aspects of VTS and VTP as given in this Standard.

2.2.1 Data Structure Definition (DSD)

The DSD defines how the cells of the CDS are organized for the purposes of access, addressing, and control.

The DSD also holds the parameters negotiated and accepted between the communicating p-users, for example:

- class of service currently applicable, (which determines the meaning of much of the remaining parametric information),
- global parameters for interpretation relating to the object definition (allows the correct interpretation of the content of the cells of the CDS),
- access control facilities negotiated (number and scope of access control tokens, etc.),
- other class-dependent parameters, e.g.:
  . pointer to current Cell and/or Form or other complex structure being addressed,
  . current attribute(s) in use and defaults,
  . object attributes allowed, permissible values of each and default values,
  . etc.

The DSD therefore holds all the necessary presentation environment information of the VTS as it is currently being used between the communicating p-users. The exact way in which the DSD defines the CDS is class dependent and is defined in the documents defining the VTP standards for the different classes of the VTS.

A self-consistent complete and usable set of such parameters makes up the presentation environment (PE) for VTS. The GPS according to the negotiated subset, etc., may provide facilities for use of multiple PEs (one at a time) with negotiation and re-negotiation of PEs. The VTS user does not, in general, call the GPS subset and facilities directly but these are derived from the VTS parameters requested by the VTS users at establishment time and possibly at other times.

2.2.2 Conceptual Data Store (CDS)

The CDS provides the conceptual storage cells for the Atomic Objects, mutual access to which is the main purpose of the activity of the communicating p-users in the context of the VTS. The CDS consists of a finite or (potentially) infinite set of storage cells each capable of holding (conceptually) one complete Atomic Object, however complex the definition of the Atomic Object may be. The storage cells are independent of each other except where a relationship is defined in the DSD.

An Atomic Object must have at least one component but may have additional components:

- Primary Value : This is the essential data content of the cell,
- Object Attribute Values : These parts of the Atomic Object hold the particular values for that object of each Object Attribute which has been defined as applicable to the CDS in use. More information on the important topic of Attributes is given in the following sub-clause.

2.2.3 Attributes

The Attributes information in the CCA is an important part of the operation of VTS. Attributes fall into two major classes:

(a) Object Attributes which are, at least conceptually, contained within the individual Cells of the CDS so that they form part of the Atomic Object in each Cell and qualify the Primary Values of these objects (the fact that some or all such attributes may, in a particular case, be able to be held on a once-per-CCA basis does not alter this classification or the way in which their use is defined).

(b) Global Attributes which are contained outside the CDS because either they are inherently applicable to the whole CDS (for example those which define grouping of cells) or are related to other components of the CCA; sub-classifications may arise from the requirements of advanced classes and may be added to a later version. In particular a hierarchy of global Attributes is likely to be necessary similar to the CDS Structuring capability.

In both classes some attribute values are agreed at establishment time or at negotiation time and are then fixed throughout transfer-enclosure. For others service facilities will be provided to allow the values of the attributes, and possibly the scope of applicability, to be changed explicitly. Some attributes may also be affected implicitly by other activities. These two characteristics of attributes, orthogonal to the above classification, are known as Constant and Variable, and can qualify the above classification. This characteristic may not be inherent in an attribute definition but depend on how the attribute is brought into use, i.e. an attribute which, in itself, allows variability, can be fixed at establishment time and thus be unchangeable for the life of the p-connection.

The range of values that may be given to a Variable Attribute is fixed at negotiation time.

Ability to alter Attribute Values is subject to Access Control; this may or may not be related closely in particular cases to the access control on the CDS cells. For example, in Basic Class, manipulation of the Attribute Values of a cell or cells of the CDS is always conditional on having write access to the (whole) CDS, this being the form of access control defined in that class.

2.2.4 Access Control (AC)

The AC provides the information whereby the access by each p-user to the contents of the other parts of the CCA can be controlled. The complexity of this control, defined for a class, and negotiated for a particular presentation-connection, can vary widely. Access Control applies, in general, to the right to examine as well as the right to change, and in appropriate cases these can be separately controlled.

This conceptual mechanism can range from the simple two-way alternate switch on the CDS and DSD (such as is used in Basic Class of VTS) to a very sophisticated locking mechanism with a fine "granularity" on the data in the CCA, should this be appropriate for a particular class. Separate control over read-access as distinct from read-and-write access may also be appropriate in certain cases. Inclusion of a delivery-control facility with the concept of a "delivery unit" enables an intermediate form of control in which an update and the ability to examine it is passed but the right to update remains with the sender.

The concept of tokens is used for the description of the Access Control mechanisms. A token is an abstract thing with which is associated (by an implicit or explicit assignment process) some action on the Virtual Terminal and which mediates the right to initiate this action between the two p-users. At any particular instant of the operation of the Virtual Terminal, i.e. during the life of a presentation-connection, each defined token is in the possession of at most one of the presentation users and this user, and this user only, can initiate the associated action. Facilities are provided in the Service for the transfer of a token from the current owner to the peer p-user. The p-service will police the primitives issued by the p-users for conformance to the current state of the defined tokens. Some tokens are related to the GPS facilities used and others are related to the requirements of the PE/CCA; further information is given later.

2.2.5 Control Signalling and Status Store (CSS)

The CSS Store is a collection of aeras within the CCA which is used as a repository for Control, Signalling

and Status information.

CSS Areas may be separately declared during negotiation and include parameters concerned with Access Control, Token Use and Data Structure. Once declared a CSS Area may be attached to one or more Virtual Devices and may be used for one or more purposes as determined by the mechanism realization parameters.

The separate declarations of CSS Areas are in addition to those CSS areas automatically brought into existence as a result of the agreement on each Virtual Device.

2.2.6 Virtual Devices

Following negotiation of the CSS areas, one or more Virtual Devices will be agreed for attachment to the CCA. A Virtual Device corresponds to the logical requirements of the service user. As part of the negotiation process a number of parameters are available to control the mapping of these logical attributes such as emphasis levels, colour and of layout control, onto the real device.

As a result of the negotiation of a Virtual Device, a minimal CSS Data Area is automatically brought into existence to allow basic device control to be performed. In general this is class dependent but would typically include the following:

- ON/OFF control
- Interrupt
- Status Alert
- Alarm

In addition to the above default CSS Area, a number of previously declared class dependent CSS Areas may be attached to the Virtual Device to provide more sophisticated Control, Signalling and Status mechanisms as described below.

How they are used is specified by means of the CSS realization specification parameters.

Signalling

A CSS area can be used with appropriate realization parameters as a repository for signalling information attached to one or more Virtual Devices to represent the following:

- Attention Keys
- Tracker Ball
- Light Pen
- etc.

Status

A CSS area can likewise be used as a repository for status information.

Control

A CSS area can likewise be used as a repository for control information. This provides a method for device control either in the explicit ON/OFF sense or for a more flexible user-defined technique.

2.3 Service Description

2.3.1 Introduction

The ECMA Virtual Terminal Service (VTS), as a service of the Presentation Layer of the ISO OSI Model, operates within the generic framework of the Generic Presentation Services (GPS) supported by the ECMA Generic Presentation Protocol (GPP). Many aspects of the overall service description are therefore by reference to GPS, with additional detail of parameters as necessary. The actual usage of the services included in the total GPS is dependent on the actual Class of VTS and the requirements of the users within that class and may be simple in some cases.

The GPS (and the supporting GPP protocol) are not a formal sub-layer of the p-layer but rather a set of facilities which are used as required by the VTS functionality. Thus VTS uses the GPS facilities to establish the p-connection with the correct peer p-user, set up the various service parameters needed to determine the p-layer operations, including the characteristics of the underlying session service, enable negotiation and re-negotiation of the presentation environment(s) to be used on the p-connection, and control the use of these from time to time by the p-users.

The GPS gives the model of Presentation Layer within which the VTS services are accessed by p-users.

2.3.2 Functional Phases of the Virtual Terminal Service

In general the VTS uses all the functional phases of the GPS, and has no special phases not subsumed within these. Particular instances of VTS may not use the Negotiation Facility if the p-users do not require or permit negotiation except at establishment time, i.e. a single presentation environment (CCA) is set up at this time and used unchanged until disestablishment occurs.

The choice of GPS/GPP subset, see ECMA-86, is not made explicitly by the p-users but is implied in the choice of other p-service parameters.

This Generic VTP Standard does not define subsets of the Generic VT Service. Indication is given in later subclauses of clause 2.3, of the major aspects of the

service which are optional insofar as individual VTS Class standards may exclude their use or make them subject to negotiation by the VTS-users. These aspects are largely related to the subsets defined for GPS in ECMA-86.

Individual VS Class standards will define any formal subsets for the class and how these affect the choice of the major GVT and/or GPS aspects as above.

2.3.3 Establishment and Termination of P-Connection for VTS

The general description of these phases given in GPS is applicable to GVT. Additional VTS specific information is given. In some cases indication is given of additional information specific to particular VTS Classes which will be supplied by individual class standards.

2.3.3.1 Establishment of Presentation Connection

The service element applicable is as follows which also gives the VTS specific parameter usage. Aspects not mentioned conform to the description in GPS.

P-CONNECT

Parameters with VTS specific usage:

p-max-pes :
p-save-pe-capab :
p-fixed-pe-params :
p-initial-pe-params :
} see additional information.

The component parameters of these are used as follows:

p-service-ident: Applicable value is "VTS".

p-service-class-ident: This takes one of the symbolic values defined for the Classes of VTS, for example "basic" to call for the provisions of Basic Class VTS standard.

p-service/class-version-idents: Applicable if p-service-class is selected and enables the class version to be agreed. Applicable values are as given in individual VT class standards which may also define a default value. With this version the service class and class versions must be included in p-fixed-pe-params and are thus not subject to change during the life of the p-connection. A later version is expected to remove this restriction and allow a re-negotiation of the class, with applicable version.

p-service/class-subset-ident: This is always a class subset and is used as to be described in individual class standards; there are no GVT subsets.

p-pe-profile: Optional: used as to be described in individual VTS Class standards.

p-special-conventions: Optional: used as to be described in individual VTS Class standards.

In this version, in accordance with the restriction in GPS, the session service parameters are selected at establishment time, being implied by parameters in p-fixed-pe-params, and are not changed during the life of the p-connection.

Additional information

General service tokens defined by the above parameters (see 2.3.4) will be owned initially by the issuer of the P-CONNECT request primitive. Tokens specific to a PE will be owned as defined for the particular cases. Some classes and subsets may make use of multiple-pe and pe-save options; value of parameter p-max-pes is relevant while such class/subset is in use but a value can be agreed independently of such use.

2.3.3.2 Termination of Presentation Connection

The following service elements are relevant to VTS and are as described in GPS with VTS specific information as below:

P-RELEASE,

P-DISCONNECT,

P-ABORT.

VTS specific information

The issue of P-RELEASE request primitive is always mediated by the p-terminate-token. The Release is, however, not always "negotiated" (i.e. may be unconditional) as this depends on the session mapping for the individual class standard.

2.3.4 Token Control Facilities

GPS includes no token control service elements as the nature of usage of token facilities is specific service-dependent. VTS includes a common set of service elements for the control of the service tokens which control the activities in various phases and enclosures of the VTS p-service. A list of types of service tokens available in the general case is as follows:

- p-terminate-token, - enclosure-token : general service tokens
- negotiate-token : used in negotiation-enclosure
- transfer-token : used in transfer-enclosure
- write-access-token : used in transfer-enclosure

This is not an exhaustive list. The actual requirements for tokens in transfer-enclosure will vary with the class but the transfer-token is the minimum provision. Some classes and presentation environments may define other tokens required when transfer-enclosure is using such a

PE. A write-access-token is an example of such a token which will very frequently be needed. Some classes will combine the above token functions into a smaller number of actual tokens and this may render some of the service elements and/or parameters to them redundant. Thus a requirement for a write-access-token will frequently map this with the transfer-token, but the formal service tokens will be kept distinct.

The VTS specific service elements for token control are as given below.

P(VT)-GIVE-TOKENS

Purpose: To enable a p-user to pass ownership of one or more service tokens, currently owned by this p-user, to the peer p-user.

Structures: Non-confirmed, type 1, RI.

Parameters:

Parameter Name	req	ind
p-token-ident	D(n)	U(n)
p-encl-service-specific	D	U

Parameter description

p-token-ident: This parameter is required when there is more than one token with separate token control, and will designate which of them are given. See also Additional Information.

p-encl-service-specific: Optional: applicable if a token of the transfer-enclosure is being given and allows class and PE specific sub-parameters to be included. Availability and use of this parameter is class dependent; no current class makes use of it and a later version may withdraw it.

Usage and Effects

Request primitive may be issued at any time by the current owner of the designated token(s) unless excluded by specific services. Effects are sequenced and non destructive.

The p-service will record any change of ownership of tokens. Effects will be null if none of the designated tokens are currently owned by the issuing p-user.

Additional Information

A token may be designated in p-token-ident only if this is valid in the current state of activity, as follows, and not excluded by the class standard:

- p-terminate-token: valid at any time,
- enclosure-token : valid at any time,
- negotiate-token : valid only when a negotiation-enclosure is in progress,
- transfer-token,write-access-token and any other token specific to a PE: valid only when transfer-enclosure is in progress.

Separate control over some or all tokens is VT Class dependent.

P(VT)-REQUEST-TOKENS

Purpose: Used by a p-user who wishes to gain ownership of one or more service tokens not currently owned so as to be able to initiate the associated token-controlled activity(ies).

Structures: Non-confirmed, type 1, RI.

Parameters:

Parameter Name	req	ind
p-token-ident	D(n)	U(n)

Parameter description

p-token-ident: This parameter is required when there is more than one token with separate token control, and will designate which of them are requested. See also Additional Information.

Usage and Effects

Request primitive may be issued at any time by either p-user except under circumstances excluded by specific class standards. Effects are sequenced and non-destructive.

There is no recorded effect on the p-service itself. Effects may be null (i.e. no indication) if all the designated tokens are owned by the issuing p-user.

Additional information

A token may be designated in p-token-ident only if this is valid in the current state of activity, as follows, and not excluded by the class standard:

- p-terminate-token: valid at any time,
- enclosure-token : valid at any time,
- negotiate-token : valid only when negotiation-enclosure is in progress,
- transfer-token, write-access-token and any other token specific to a PE: valid only when transfer-enclosure is in progress.

Separate control over some or all tokens is VT Class dependent.

2.3.5 Enclosure Control Facility

The service elements below are applicable to this facility and are as described in GPS with VTS specific information as follows. Further information is given in the following clauses and will be given in individual class standards.

P-CHANGE-ENCLOSURE,

P-CLOSE/ABORT-ENCLOSURE.

VTS specific information

The enclosure-token is always defined in VTS and mediates the issue of P-CHANGE-ENCLOSURE request primitive.

2.3.6 Negotiation Facility and Overview of VTS Parameters

The Virtual Terminal Service of the Presentation Layer cannot perform a useful function for the p-users until a self-consistent set of values of parameters and attributes has been established to make up a presentation environment (PE). For a particular p-connection at least one Defined PE must exist before Transfer-enclosure can be entered. Depending on the Class of VTS and user requirement characteristics it may be possible to have a number of PEs defined at any one time, i.e. Multiple-PE option in use.

The Initial PE established at the time of establishment of p-connection may be complete, i.e. a Defined PE, and thus usable, or incomplete.

Where the VTS users require it, VTS provides, by means of the service elements described in GPS a negotiation facility which enables a presentation environment to be agreed by the two p-users. This negotiation service is concerned with any parameter in the presentation layer which affects the p-users usage of the VTS. Both levels of negotiation facility in GPS are available in VTS, i.e. single-interaction negotiation and multiple-interaction negotiation in negotiation-enclosure.

2.3.6.1 Single-interaction Negotiation

The service element provided by GPS for this level of negotiation facility is applicable as given below, which also gives any additional VTS specific information. Further specific information may be given in individual class standards.

P-PERFORM-NEGOTIATION

VTS specific parameter usage

p-pe-ident, p-draft-pe; in general VTS allows the use of multiple presentation environments and these parameters are then relevant as described in GPS; particular classes of VTS may restrict use of Multiple-PE option and this is also a selectable parameter, and this may affect usage and applicable values of these parameters.

p-pe-specific-params; these parameters are VTS Class specific and usage and values will be defined in individual class standards. Reference should be made to the description of P-CONNECT for VTS specific rules for the "fixing" of certain parameters which GPP would otherwise permit to be used in these parameters.

Other parameters are as in GPS.

Additional information

Issue of request primitive is mediated by enclosure-token from null-enclosure and transfer-token from transfer-enclosure. The token remains with the initiating p-user after the service element.

2.3.6.2 Multiple-interaction Negotiation and Negotiation-enclosure

The service elements provided by GPS for this level of negotiation facility are applicable as given below, which also gives any additional VTS specific information. Further specific information may be required in class standards.

P-CHANGE-ENCLOSURE

This is used to enter and exit from negotiation-enclosure as described in sub-clause 2.3.5.

VTS specific parameter usage

Entry parameters:

p-pe-ident, p-draft-pe; see 2.3.6.1

p-initial-pe-params; these parameters are VTS Class specific and usage and values will be defined in individual class standards. Reference should be made to the description of P-CONNECT for VTS specific rules for the "fixing" of certain parameters which GPP would otherwise permit to be used in these parameters.

Exit parameters:

p-pe-action; as GPS.

Other parameters are as in GPS.

Additional information

The (single) negotiate-token is implicitly defined in negotiation-enclosure and is initially owned by the issuer of the P-CHANGE-ENCLOSURE request primitive.

P-CLOSE/ABORT-ENCLOSURE: no VTS specific information.

Facilities within negotiation-enclosure:

P-NEG-INVITE

P-NEG-OFFER

P-NEG-ACCEPT

P-NEG-REJECT

VTS specific parameter usage

p-param-ident-list, p-param-list; these parameters are VTS Class specific and usage and values will be defined in individual class standards. Parameters not mentioned are used as in GPS.

2.3.6.3 Overview of Presentation Service Parameters for VTS

The number and nature of the presentation parameters relevant to a particular use of the presentation service are service type and class-dependent to a very large extent. However the general principles of the negotiation service as described in GPP are of general application.

Some cases may not require the full generality in practice and the use of the service (and the protocol) may in such cases degenerate to a small subset of the full capabilities. An example of this is where a Basic Class VTS requires only one of the standard profiles defined for that class of VTS and this can thus be selected and accepted at connection establishment time during execution of P-CONNECT service element with no further use made of the negotiation service (and protocol) itself. In such a case there could be no negotiation capability at all, the presentation environment being fully complete, and unchangeable after the connection establishment.

In other cases, where the values of some presentation parameters cannot be negotiated until certain others have been given agreed values, the service elements within a negotiation-enclosure may be used several times, the set of parameter values forming the final presentation environment being built up stage by stage. Different "threads" of the parameter structure may be negotiated concurrently and be at different stages at any one time.

To simplify the negotiation process the concept of profiles will usually be available. A profile is a complete (i.e. usable) set of p-connection parameters; in some cases there can be associated mandatory parameters to accommodate common variations of a profile. Profiles are VTS-class specific and will be defined in individual class standards. Individual service elements indicate whether a profile is a valid parameter and whether further

negotiation based on the profile can be performed in the same service element.

2.3.6.4 Directed Graph Structure of Presentation Parameters

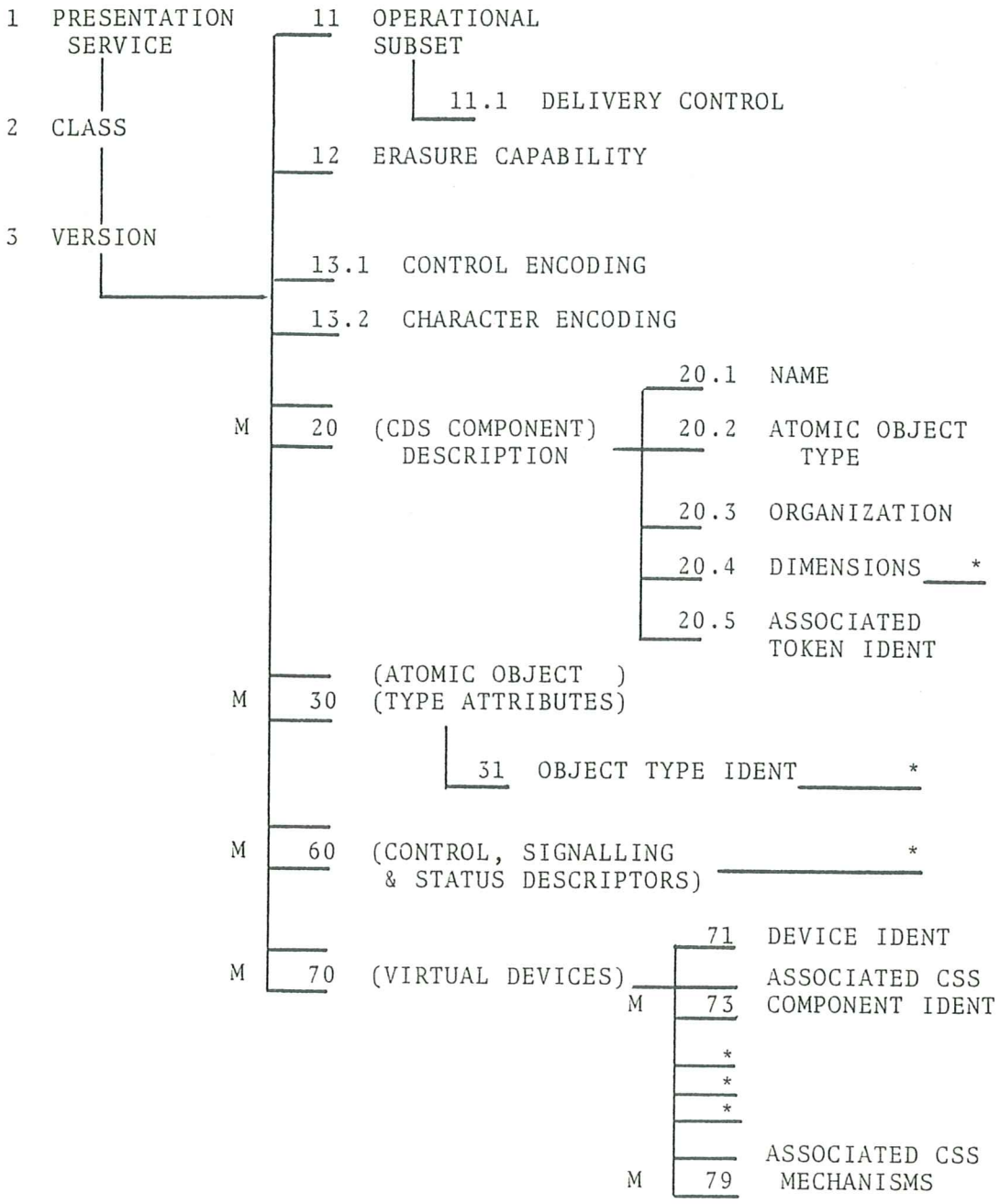
GPP introduces the concept of the directed graph of presentation parameters as the method of showing the interdependency relationships between any particular parameter and others higher or lower in the graph.

Individual VTS Class standards will define the directed graph for all parameters relevant to the standard and where appropriate indicate the dependence of this graph on a higher level graph. For example the graph of presentation parameters specific to Basic Class of VTS is dependent on a higher level graph of the general parameters of presentation service. A graph definition may group together certain sets of parameters with the implication that these cannot be quoted individually at any point in the negotiation process.

GPP gives the rules for the use of the negotiation facility services to build up the definition of a Presentation Environment (PE) from some starting point.

2.3.6.5 Skeleton Directed Graph for Parameters

Specific Class Standards will select from, and extend, this skeleton graph and each will show a complete self-consistent graph.



M indicates that multiple occurrence is permitted.

( ) enclose parameter names which have no value themselves but serve a grouping purpose for dependent parameters.

* this skeleton graph does not attempt to indicate all positions at which extensions for class dependent (sets of related) parameters may be added - some of the obvious ones are indicated by *.

Nodes may be omitted in individual class standards if they are redundant for any reason.

2.3.7 Information Entry and Presentation to Other P-user

The concepts of Transfer Enclosure, Enclosure Control and Multiple-PE operation in GPS are applicable to VTS in the general case although individual VTS Class Standards may exclude the use of multiple-pe option.

The following material gives some VTS aspects of this part of the p-service. Each individual VTS Class Standard will need to build on this general information with the detail applicable to its methods of operation. Those aspects which are defined herein in some detail are expected to be applicable to all or at least most classes. Each Class Standard will indicate applicability of any parts of this general information and give any variations and further information as necessary to complete the definition for that Class.

2.3.7.1 The Transfer Enclosure

The generic description in GPS is applicable to VTS. Thus entry can be explicit from the Establishment Phase, and exit can be made directly to the Disestablishment Phase (under token control for orderly exit). The use of P-CHANGE-ENCLOSURE enables explicit control over enclosure transitions, in particular allows use of the negotiation-enclosure where this is an agreed facility, and also allows the selection of PE in cases where multiple-pe option is in use.

2.3.7.2 Enclosure Control Facility

The service elements below are available for enclosure control, including control of entry to, and exit from, transfer-enclosure, and conform to the description in sub-clause 2.3.5. There is no GVT specific information; the use of the entry and exit parameters is subject to class and class-subset definition and will be given in individual class standards.

P-CHANGE-ENCLOSURE

P-CLOSE/ABORT-ENCLOSURE

2.3.7.3 Access Control Facilities

Access control is concerned with regulating the manipulation by the two p-users of parts of the CCA. In general, for any particular regulated part of the CCA, at any one time at most one p-user will have the current ability to alter (update) the contents, either "data" or attributes. At times when neither has the right to update, a particular p-user may or may not have the right to read (examine) the part. A general rule relating read and write access is that if one p-user has write access then the peer p-user cannot have (formal) read access (although there may actually be no formal interlock policed by the p-service).

The Access Control Facilities of the p-service are described using the concept of tokens as introduced in sub-clause 2.2.4.

Within transfer-enclosure the type of token relevant to access control is known as a write-access-token, and one or more such tokens will generally be used in a particular CCA depending on the provisions of the Class of VTS and the requirements of the p-users as agreed during establishment and/or negotiation, i.e. as expressed in the PE. Each such token in use will apply to a particular subset of the total CCA, (its scope can be the whole CCA), and gives the current owner the ability to examine and to update the associated parts of the CCA. In many cases the transfer-token will be used to provide the function of one of or the only write-access-token but the functions will be kept formally distinct. Some PEs may define that write access is free of token control for some parts or all of the CCA.

The minimum provision of tokens within transfer-enclosure is the transfer-token which will be implicitly assigned (to the then current owner of the enclosure-token) when transfer-enclosure is entered.

Frequently the transfer-token will also be used as a write-access-token. This is the default case. If there is requirement for additional tokens in the class and PE these tokens and their scope of control will be defined implicitly or explicitly at or immediately after the entrance to transfer-enclosure, including the initial owner, similarly for a case where write access is token free.

Where the service facilities of a Class so require, the capability for further write-access-tokens with implicit or negotiable partial scope within the CCA will be provided in that class standard as part of the definition of the presentation environment, including, where appropriate, negotiable parameter(s). The general facilities described in 2.3.4 will be used for token manipulation.

2.3.7.4 Delivery Control

Delivery control is a facility related to the more rigorous forms of access control administrered by the token mechanisms but has some rather different characteristics. It will usually be used as a supplement to the formal write access control given by one or more write-access-tokens but where there is no provision for formally controlled read access control by read-access-tokens. Its use is part of the presentation environment definition. The method by which use of delivery-control is agreed is class specific.

Delivery control allows the owner of a write-access-token who has made some updates to (the relevant parts of) the CCA to determine when these updates are actually delivered to the peer p-user instead of leaving this to the discretion of the p-service, in circumstances where it would not be appropriate to give write access to the peer p-user. The service indications of updates are then given to the peer p-user only at the point in the series of updates at which the deliver request was made, thus enabling a simple form of synchronization between the two p-users. "Net effect" applies between delivery points (subject to detailed definition of certain forms of update). An extension to the simplest form of the deliver facility enables a p-user to solicit acknowledgement that the peer p-user has received the update. Where such acknowledgement is requested, which can be selective on the basis of designated (owned) write-access-tokens, the update permission (and ability to issue P(VT)-DELIVER request primitive) normally implied by each such token is suspended until the relevant acknowledgement (which can also be token selective) has been received. When acknowledgement is not requested there is no provision in the p-service against the issuing p-user making further updates to any part of the CCA for which an applicable token is owned.

The delivery feature, including receipt acknowledgement, can be used without delivery control being in force but delivery may then also occur at other times at the discretion of the p-service.

2.3.7.5 Entry of Information to the CDS

This is concerned with the entry by one p-user of information into the CDS for subsequent visualization of the resulting content of the CDS by the other p-user as the conceptual image. Information comprises atomic object primary values and attribute values.

The next storage location at which data is to be entered is marked by a pointer. A location so indicated is known as an Active Location. The capabilities for explicit moves of an active location are determined by the facilities of the class and the negotiated parameters of the PE. The possibility of multiple pointers/active locations is not excluded. The pointers may also apply to the entry of attribute values.

A service element is provided as below for the entry of information into the CDS; the standard generic description is simple but the associated parameters are class dependent and will frequently be complex.

P(VT)-DATA

Purpose: To pass to the p-service information to be entered into the CDS.

Structures: Request only, type 4, RO, or
Non-confirmed, type 1, RI, see effects below

Parameters:

Parameter Name	req	ind
p-cds-data	D	U

Parameter description

p-cds-data: Conveys the CDS update information see sub-clause 2.3.7.7.

Usage and Effects

For classes in which one or more write-access-tokens are defined, this request primitive may be issued only by owner of at least one write-access-token. Effects are sequenced with respect to all other sequenced service elements, and non-destructive.

Causes the p-service to adjust the contents of the CDS according to the value of p-cds-data. P(VT)-DATA indication primitive will not be given if delivery control is in use; if not in use the packaging of updates into indications is at the discretion of the p-service. Logical sequence significance of updates will be maintained also the sequence with other sequenced service elements.

Additional information

Description of the possible contents of the p-cds-data parameter is given in sub-clause 2.3.7.7. The request primitive is Invalid if p-cds-data contains update to any part of the CDS not within the scope of the write-access-tokens owned by the issuing p-user.

2.3.7.6 Presentation to the Other P-USER

A frequent case of operation of the VTS will be for one p-user currently having write access to all or part of the CCA (CDS and/or CSS areas) to make a series of updates, and when these are completed to request that these are to be made available to (are delivered to) the peer p-user. The issuer of an update may request acknowledgement from the peer p-user that receipt has occurred. Delivery control allows delivery (with solicit of acknowledgement if available) to be made at precisely determined points without passing write-access at the same time.

The issuer of the updates can optionally yield some or all of the write-access-tokens currently owned. This implies delivery of any updates for parts of the CCA covered by the relinquished tokens. The acknowledgement feature is not available when write-access is released.

The general mechanisms providing the above facilities are described in sub-clause 2.3.7.3.

There is no explicit primitive in this service description whereby a p-user can call for information from the CCA. The supplier of CCA update information, always the owner of a write-access-token to some part of the CCA, controls when updates are made available to the peer p-user, and, except when write-access is relinquished can, by means of the acknowledgement feature, request to be advised when the peer p-user has received the updates.

Below are given the generic facilities available in VTS for control of presentation. A particular class may not provide this full generality, either in provision of primitives or use of parameters.

The above general information applies to the CDS and to CSS areas.

P(VT)-DELIVER

Purpose: To cause the p-service to deliver all outstanding CCA updates relating to one or more write-access-tokens and optionally, solicit acknowledgement of receipt.

Structures: Non-confirmed, type 1, RI.

Parameters:

Parameter Name	req	ind
p-token-ident	D(n)	U(n)
p-solicit-ack	D	U
p-cds/css-data	x	U(n)

Parameter description

p-token-ident: Designates the write-access-tokens relating to which the delivery action is required, see Effects. Not required if there is only one write-access-token. Default is "all" tokens.

p-solicit-ack: Optional: enables the issuer of the request primitive to solicit an acknowledgement (by P(VT)-ACK-RECEIPT). Symbolic values are "yes", "no"; default is "no".

p-cds/css-data: Supplied by p-service to indicate the cds/css-update information available, see 2.3.7.7.

Usage and Effects

Request primitive can be issued only by the owner of the token(s) designated in the request (by implication, there can otherwise have been no updates). Effects are sequenced and non-destructive.

It causes any updates for the CDS or CSS areas held within the p-service relating to the designated write-access-tokens to be made availabe to the peer p-user. According to the value of p-solicit-ack, the update capability implied by posession of the designated token(s) is suspended until P(VT)-ACK-RECEIPT indication primitive has occurred, see P(VT)-ACK-RECEIPT below.

P(VT)-GIVE-TOKENS

This service element is not specific to operation within transfer-enclosure but specific use is made of the parameters. The general description in 2.3.4 is applicable.

Specific detail for transfer-enclosure: Use is made, optionally, of the p-encl-service-specific parameter; the usage is class and subset dependent. Deliver action is implicit in this service element.

P(VT)-ACK-RECEIPT

Purpose: To acknowledge receipt by a p-user of update information made available as a result of an occurrence of P(VT)-DELIVER service element.

Structures: Non-confirmed, type 1, RI.

Parameters:

Parameter Name	req	ind
p-token-ident	D(n)	U(n)

Parameter descriptions

p-token-ident: Designates the write-access-tokens associated with the updates which are being acknowledged, see Effects and Additional information. Not required if there is only one write-access-token. Default is "all" tokens defined for the PE in use, see Additional information.

Usage and Effects

The request primitive will be issued by a p-user only after a P(VT)-DELIVER indication primitive has been issued

on the service interface with p-solicit-ack parameter present. Effects are sequenced and non-destructive. It releases the suspension of the write-access-tokens designated in the request primitive, see P(VT)-DELIVER, and allows processing of further service elements relevant to these tokens.

Additional information

Inclusion in the request primitive of a write-access-token which is not currently associated with a request for acknowledgement (i.e. is not suspended) is Invalid. Thus "all" default is valid only when all defined tokens for the current PE are suspended.

2.3.7.7 CCA Update Parameters

Sub-clauses 2.3.7.5 and 2.3.7.6 describe the p-service elements which are available in transfer-enclosure (subject to the PE definition) for updating the CDS and making the updates available to the peer p-user. The parameter p-cds-data is provided in the P(VT)-DATA request primitive and in a number of indication primitives. This is a compound parameter and the components will, in general, be a mixture of position controls, atomic object primary values and attribute values and extents.

It is not possible to give in this general VTS service description any detailed description of the contents of these parameters in particular cases as these are class-dependent. It is likely that a class will provide sub-parameters for the following general types of CDS update information:

a) OBJECT-PRIMARY-VALUE: Conveys items of nature object-primary-value according to the definition of the atomic object for the class.

b) POSITION-CONTROL: Conveys items concerned with adjusting pointers to control the action of subsequent OBJECT-PRIMARY-VALUE or ATTRIBUTE items; position control facilities depend on the definition of the CDS for the class. Individual class definitions will state whether it is possible to move the pointer to a value which is outside the scope of a write-access-token owned by the issuer of the P(VT)-DATA request primitive but it is always invalid to attempt to enter primary values or attribute values at such a point.

c) ATTRIBUTE: Conveys items concerned with adjusting the current values of the attributes (i.e. of those attributes which have been agreed as being variable in a transfer-enclosure). Further notes on attribute updating are given below.

2.3.7.8 Attribute Update Parameters

Control of the Attribute information in the CCA is an important part of the operation of a VTS. A broad classification of Attributes is given in sub-clause 2.2.3.

With all types of attribute some attributes are only manipulated at negotiation time and are fixed during transfer-enclosure. For attributes which are variable during transfer-enclosure the P(VT)-DATA service element is available to allow the values of the attributes, and possibly the scope of applicability, to be changed explicitly. Some attributes may also be affected implicitly by other operations of the service.

Sub-clause 2.2.3 also gives general aspects of access control as it relates to attributes. In general, attributes are controlled by write-access-tokens as are various other parts of the CCA and the flexibility available is class dependent. For example, in Basic Class, manipulation of the Attributes Values of a cell or cells of the CDS is always conditional on having Write Access to the whole CDS, this being the form of access control defined in that class. In a more complex class, particular sets of attributes, possibly with a hierarchical structure, may be defined as having an access control token associated with them, so that the right to alter them may be controlled independently of other elements of the CCA. It is also important not to confuse the right to alter the current value of a negotiated attribute within some negotiated range (which can be done within transfer-enclosure) with the right to renegotiate the range itself (which has to be done by a negotiation operation).

2.3.8 Control, Signalling and Status (CSS) Areas

An arbitrary number of named areas within the CSS may be negotiated in addition to any which are created by default for each virtual device. These areas are distinct from the CDS and in general subject to different access control rules. Their use may imply a delivery and the passing of ownership of a token.

This Standard does not define such areas precisely - each individual class standard will describe the possibilities it offers for definition and use of CSS areas. This Standard can only give the general concepts underlying the definition and use of CSS areas for control, signalling and status.

Each declared and default CSS area has the following negotiated or defaulted service characteristics (attributes):

- name,
- subject to access control (and associated token-ident),
- whether it is a CSS area the use of which implies the release of the associated token (and delivery of any information to which these controls access). (For convenience of reference this feature is termed a "trigger" mechanism),
- data format and length.

In the case of a "declared" CSS area "name" is a user provided name - such a declared CSS area may be associated with one or more virtual devices for one or more purposes. In the case of a default CSS area the name is the same as the user supplied virtual device name and accordingly such a CSS area is unique to the virtual device with the same name.

Permitted values for "access control" are "yes" with the associated token-identifier, or "no".

If "yes" then information can only be written into the CSS area by the p-user in possession of the specified token, using P(VT)-CSS, see sub-clause 2.3.9, and such information is subject to the same delivery control as any CDS data controlled by the same token. The special case where the CSS area is also a "trigger" mechanism is covered below.

If "no" then logically the information placed in the CSS area by either p-user at any time, using P(VT)-FREECSS and independent of the possession of any token(s), is assumed immediately delivered.

If the CSS area is a "trigger" mechanism (only valid if subject to access control) then the entry of data into it implicitly invokes a "deliver" and a "give token" for the associated token. Such a facility allows "reverse break", "attention", "interrupt", "function key", "light pen", as well as pre-emptive control facilities to be provided.

Data format and length are dealt with by the specific class standards, there being no a priori reason why any general structured data area should be defined and used.

Default CSS areas are simple primitive areas providing the minimum which might be considered meaningful for devices of the associated class. Their existence (and therefore implementation overhead) is not negotiable - their characteristics may be. For example Basic Class offers a one octet area which allows on/off control, interrupt, status alert and alarm only.

2.3.9 Virtual Devices

One or more virtual devices can be negotiated as part of a presentation environment. This GVT Standard does not define the nature of these devices in detail only in terms of certain general characteristics and capabilities. Individual class standards will cover relevant details.

Each virtual device will have some or all of the CDS associated with it (those areas of the CDS associated with a particular token can also be associated with one or more devices), will have a simple CSS (control, signalling and status) area associated uniquely with it by default, and may have one or more declared CSS areas also associated with it. Such declared CSS areas may be used for control and/or signalling and/or status information for one or more virtual devices, and, in general, may be "controlled" by a specified token or "uncontrolled", i.e. not subject to any token.

In addition to specifying associated CDS and CSS areas each virtual device negotiation can specify device attributes, e.g. in, out or in/out, and agree realization controls, e.g. for character repertoire, emphasis, foreground and background colour and layout control. The precise attributes for which such realization control is required and how it is specified will be given in individual class standards.

The general service elements P(VT)-CSS and P(VT)-FREECSS (see description below) can be used for entering information into a CSS area.

P(VT)-CSS

Purpose: To enter information into a CSS area which is defined as "controlled", i.e. subject to token control.

Structures: Request only, type 4, RO, or
Non-confirmed, type 1, RI, see effects below.

Parameters:

Parameter Name	req	ind
p-css-area-ident	D	U *
p-css-info	D	U *

* see Effects

Parameter descriptions

p-css-area-ident: Identifies the CSS area into which information is to be entered. Value must be the name for one of the "controlled" CSS areas in the current PE.

p-css-info: Conveys the information to be entered into the designated CSS area. Value must be consistent with the format and length agreed for the CSS-area-ident in the current PE.

Usage and Effects

Request primitive can be issued at any time by the p-user owning the relevant access token for the designated CSS-area.

The CSS-info is subject to the same rules for delivery as "normal" CDS data associated with the controlling token; thus a P(VT)-CSS indication primitive may not occur.

If the CSS area is defined as a "trigger" mechanism the issue of P(VT)-CSS request primitive will automatically imply a (logically following) P(VT)-GIVE-TOKEN for the associated token (and including the delivery always implied by a give token action) and will be executed accordingly by the p-service.

The effects of P(VT)-CSS are, in general, sequenced and non-destructive in themselves and with other P(VT)-CSS and P(VT)-DATA subject to the same token. It is class- and subset-dependent whether there are circumstances under which destruction may or will occur; this can depend on the CSS-area involved and/or on the CSS-data value and individual class standards will give detailed specification.

Additional information

Further information on the interaction of this service element with the delivery-control facility can be found under the descriptions of P(VT)-DATA and P(VT)-DELIVER.

P(VT)-FREECSS

Purpose: To enter information into a CSS area which is defined as "uncontrolled", i.e. not subject to any token control.

Structures: Non-confirmed, type 1, RI.

Parameters:

Parameter Name	req	ind
p-css-area-ident	D	U
p-css-info	D	U

Parameter descriptions

p-css-area-ident: Identifies the CSS area into which information is to be entered. Value must be the name for one of the "uncontrolled" CSS areas in the current PE.

p-css-info: Conveys the information to be entered into the designated CSS area. Value must be consistent with the format and length agreed for the CSS-area-ident in the current PE.

Usage and Effects

Request primitive can be issued at any time by either p-user. (The p-service provides no interlock against simultaneous updates).

The effects of P(VT)-FREECSS are sequenced and non-destructive within themselves and with respect to all other sequenced services. In the case of the cumulative services P(VT)-CSS and P(VT)-DATA which allow net-effecting, the effects of P(VT)-FREECSS may occur out of sequence with the net effects of the other two, and individual class standards will define circumstances (if any) where this loss of sequencing may also be destructive of any such net-effect information.

2.3.10 Service Exception Conditions

Exception conditions visible in the Service Description are either p-service detected or p-user advised.

Advanced classes of virtual terminal service are likely to provide capabilities for recovering from exceptions of both types as above, similar to or extended forms of the recovery facilities available to p-service from session layer. Such facilities are for further study.

Currently, the Generic Service Description includes only the P-ABORT as available to the p-service in the event of an exception condition in the p-service itself (invalid protocol, mis-use by a p-user, etc), and P-DISCONNECT as available to a p-user needing to abort the p-connection for any reason. These are described in earlier sub-clauses.

2.3.11 Diagnostic Information

The generic description of p-diagnostic parameter and contained information given in GPS, is applicable to VTS. Individual VTS Class standards will give details of usage of definitions in GPS and may add further diagnostic information within this generic framework.

3 Protocol

3. PROTOCOL

3.1 Protocol Overview

The communication between the two Virtual Terminal Service users is effected by means of a Virtual Terminal Protocol operated by the presentation-entities and defined in terms of Protocol Messages and Protocol Sequences. This Section discusses general (i.e. non-class-dependent) aspects as a framework for detailed definitions in class standards.

The functions offered by the protocol are not in general directly accessible to the service users (i.e. the p-users) but are employed by the presentation entities as required to support the services made available to the service user by means of a local service interface. There will not be necessarily a one-to-one correspondance between interface primitives and protocol elements or actions. The VTP standards define the protocol elements and rules for use, they do not define the local service interfaces.

The following material reflects the division of the operation of the presentation service into a number of phases and enclosures as defined in clause 2.3. Thus the protocol messages are divided into groups, corresponding to these major aspects of the activity on a presentation-connection (p-connection).

3.1.1 Relationship Between Conceptual Service Interface Requests and Presentation Protocol Messages

This discussion is mainly worded in terms of the use of the CDS but similar principles apply to the other parts of the CCA, but the argument is much simpler.

Referring first to the Model View as depicted in Figure 1, the p-users have agreed a common view of the Conceptual Communication Area. It is convenient to take as an example Basic Class which has a 1, 2 or 3 dimensional array of cells in the CDS, each cell capable of holding a character and an agreed set of character attribute values. The principles established are applicable to the general case.

With this simple case, which is all the Basic Class offers, a very simple minimal set of primitive conceptual service interface commands (function requests, etc) are all that are required for either p-user to effect any desired changes in the shared information and examine changes made by his peer p-user.

An adequate minimal set would be:

- set absolute address to any cell
- write (single) character
- write specified attribute value
- read single character
- read specified attribute value
- lock CDS
- release CDS

If each of this minimal set can be mapped into protocol messages then that set of protocol messages is functionally adequate for conveying all the necessary information concerning anything either of the p-users wishes to do.

However this is analoguous to a Turing Machine or basic assembler language approach - extremely tedious and long-winded and unsatisfactory as a description medium; equally if the protocol is restricted to the minimum required to map this basic set of functions it will be extremely inefficient.

Thus it seems inevitable that alternative commands (functions) and "macro" commands be introduced at the conceptual service interface level for convenience in description and ease of understanding and additional protocol messages for efficiency reasons. These two additions can be made independently and it is not essential that there is a one-to-one correspondence between a protocol message and an interface command. As long as there is at least one way in which a new interface command can be expanded into the basic set then the original protocol is still functionally adequate for conveying its effect.

However there may now be alternative more efficient protocol sequences which achieve the same effect. Even more importantly, regardless of whether a macro is defined at the interface level to represent them, there are certain sequences of protocol functions of the interface the net effect of which can be conveyed by a single protocol message (as an example consider ERASE-IN-LINE in terms of the basic set above).

It is not necessary or desirable in the general case that a VTP standard attempt to define a precise mapping between a conceptual service interface request and a generated protocol message sequence to convey its effect. This can and should be left as far as possible as an implementation consideration.

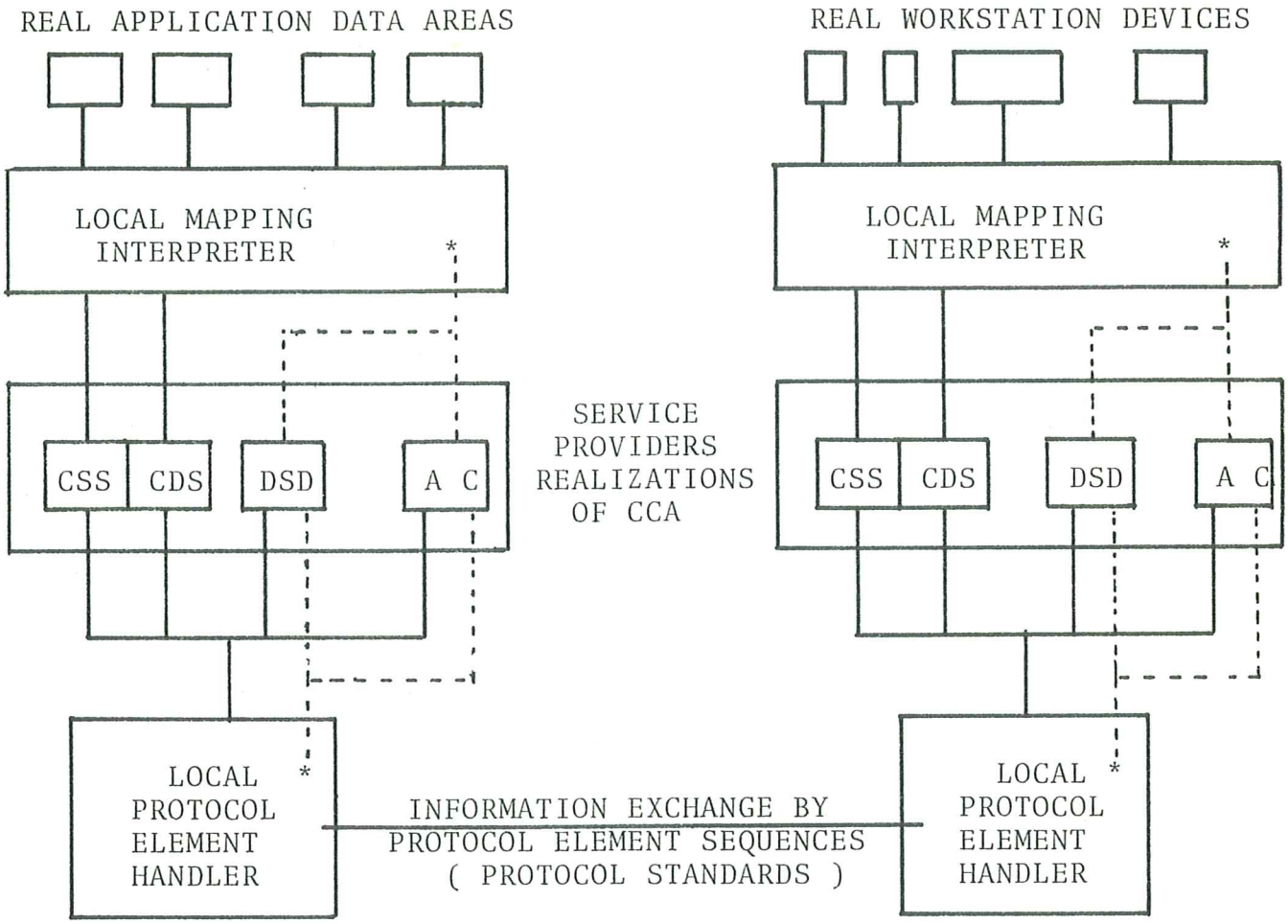


Figure 2. The Implementation View (Realization)

This last remark becomes even more pertinent when consideration is given to Figure 2. This Standard in no way constrains the local mapping or the local protocol handler/generator. As long as the net effect of a sequence of requests made across the locally defined service interface is correctly conveyed between the p-users by the generated protocol sequences, the precise mapping between them is of no concern. This leaves the implementation completely free to define its own local interfaces and to map sequences of actions on this in the best way it can onto the (subset) of defined protocol messages which the p-users have agreed to use for the particular "connection". This latter point gives rise to the possibility of the presentation entities having a negotiation on the use of protocol messages independently of the negotiation due to the presentation users requests, although the demands for protocol may arise from these requests.

3.2 Protocol Definition and Guidelines

This clause defines, mainly by reference to GPP, the Protocol Messages which are generally applicable to Virtual Terminal Protocols and the rules which determine valid sequences of such messages. It also gives the guidelines which are followed by individual class standards for the definition of protocol messages which are class dependent. Aspects of Encoding are given in clause 3.3 and of the mapping of VTP messages into Session Service primitives in clause 3.4, also mainly by reference to GPP.

Individual Class Standards will state the extent to which the general definitions in GPP and in this Standard apply and give any additional details which are needed to complete the specification of the protocol for that Class.

No formal description of the protocol is given in this Standard; individual class standards will include a complete formal description.

The protocol messages as defined in GPP are applicable to VTP as given below. Additional generic VTP specific information is given when necessary, and additional protocol messages are defined where necessary conforming to the generic principles in GPP.

3.2.1 Presentation Establishment and Termination

The following protocol messages are used in the manner defined in GPP.

PP-CONNECT-REQ (CNQ)
PP-CONNECT-RESP (CNR)

} VTS specific rules for use of certain parameters of CNQ and CNR are given in clause 3.3.

PP-RELEASE-REQ (RLQ)

PP-RELEASE-RESP (RLR)

PP-DISC (DNQ)

3.2.2 Token Manipulation

The following protocol messages are GVT specific but conform to the principles of GPP. Applicability to individual VTP Class and usage of parameters will be stated in individual class standards.

PP-GIVE-TOKENS (GTQ)

Purpose

Used to pass one or more of the protocol tokens defined in the current phase/enclosure, to the peer p-entity, either because a p-user has requested to relinquish a service token or because an internal action of the p-protocol requires it. No response is solicited as part of this protocol message group.

Contents

The parameters below are generated from the p-protocol action and/or derived from the parameters shown as relevant to P(VT)-GIVE-TOKENS request primitives;

pp-token-ident, pp-token-ident-list, pp-encl-protocol-specific.

Some parameters may not be present.

Generation

Generated by p-entity when the p-user issues a P(VT)-GIVE-TOKENS request primitive or a P(VT)-CSS request primitive to a "trigger" area, following (logically) transmission of any NDQ and/or NCQ messages conveying CCA updates relating to the tokens being passed, or when a p-protocol action requires an internal passing of a protocol token not directly visible to the p-users.

Reception

A valid GTQ designating a service-visible token(s) will cause the p-entity to give a P(VT)-GIVE-TOKENS indication primitive to the p-user, with any additional information applicable to the pp-encl-specific parameter, and record the new possession of these tokens. Where the GTQ does not designate a service-visible token the p-entity will take action appropriate to the nature of the internal protocol token(s) being passed.

Additional information

The need for token manipulation internal to the p-protocol, and the associated parameter usage, is class and subset specific.

PP-REQUEST-TOKENS (RTQ)

Purpose

Used to request possession of one or more of the protocol tokens defined in the current phase/enclosure, either because a p-user has requested possession of a service token or because an internal action of the p-protocol requires it. No response is solicited as part of this protocol message group.

Contents

The parameters below are generated from the p-protocol action and/or derived directly from the parameter shown as relevant to P(VT)-REQUEST-TOKENS request primitive; pp-token-ident, pp-token-ident-list, pp-encl-protocol-specific.

Some parameters may not be present.

Generation

Generated by p-entity when the p-user issues a P(VT)-REQUEST-TOKENS request primitive, or when a p-protocol action requires possession of a protocol token not directly visible to the p-users.

Reception

A valid RTQ requesting a service-visible token(s) will cause the p-entity to give a P(VT)-REQUEST-TOKENS indication primitive to the p-user. Where the RTQ does not request a service-visible token the p-entity will take action appropriate to the nature of the internal protocol token(s) being requested.

3.2.3 Enclosure Control Facilities

The following protocol messages are used in the manner defined in GPP.

PP-CHANGE-ENCL-REQ (CEQ)
PP-CHANGE-ENCL-RESP (CER)
PP-ABORT-ENCL-REQ (AEQ)
PP-ABORT-ENCL-RESP (AER)

Entry to negotiation-enclosure is applicable only when multiple-interaction negotiation facility is in use; class and subset dependent.

3.2.4 Negotiation Facilities

The following protocol messages are used in the manner defined in GPP. They are applicable only when negotiation facilities are in use; class and subset dependent.

PP-PERFORM-NEG-REQ (PNQ)
PP-PERFORM-NEG-RESP (PNR)

3.2.5 Facilities Within Negotiation Enclosure

The following protocol messages are used in the manner defined in GPP. They are applicable only when multiple-interaction-negotiation facilities are in use; class or subset dependent.

PP-NEG-INVITE (NIQ)
PP-NEG-OFFER (NOQ)
PP-NEG-ACCEPT (NAQ)
PP-NEG-REJECT (NRQ)

3.2.6 Facilities Within Transfer Enclosure

The following protocol messages are specific to VTS and conform to the principles of GPP. Individual VTS Class standards will state the applicability and parameter usage and may define additional protocol messages conforming to the principles given in GPP and in this Standard. Such standards will also define in detail the parameters specific to the class.

PP-DATA (NDQ)

Purpose

Used to convey CDS updates and associated parameters. No response is explicitly solicited as part of this protocol message group.

Contents

Parameters containing one or more items of CDS update information derived from one or more P(VT)-DATA request primitives.

Generation

Generated by p-entity when one or more P(VT)-DATA request primitives have been issued by the p-user. Whether or not delivery-control is in use, the p-entity will package the CDS update information due to P(VT)-DATA request primitives into NDQ protocol messages at its own discretion. (The sequence of NDQ messages due to P(VT)-DATA and NCQ messages due to P(VT)-CSS must be maintained). Issue of a P(VT)-DELIVER, P(VT)-GIVE-TOKENS, P-RELEASE, P-PERFORM-NEGOTIATION or P-CHANGE-ENCLOSURE request primitive, or a P(VT)-CSS request primitive for a "trigger" CSS area, when one or more of the associated tokens are relevant to updated parts of the CCA, will cause the sending of one or more NDQ (and NCQ) protocol messages if there are any relevant pending updates, followed by the appropriate message.

Reception

A valid NDQ protocol message will cause the p-entity to process the CDS updates; if delivery-control is not in use, a P(VT)-DATA indication primitive may be given to the p-user at the discretion of the p-entity; if delivery-control is in use no indication is given to the p-user at this time due to the NDQ protocol message itself.

Additional information

The nature and encoding of the CDS update information is largely VT class dependent and will be defined in individual class standards within the general guidelines defined in clause 3.3.

PP-CONTROL (NCQ)

Purpose

Used to convey CSS information which, due to the defined nature of the relevant CSS area, is subject to the same token control as "normal" CDS information. No response is solicited as part of this protocol message group.

Contents

The parameters below are derived directly from the parameters shown as relevant to the P(VT)-CSS request primitive:

p-css-area-ident, p-css-info.

Generation

Generated by p-entity when the p-user issues a P(VT)-CSS request primitive. Transmission of a NCQ message is at the discretion of the p-entity; sequence with other NCQ messages and with NDQ messages must be maintained. Further information under "generation" under PP-DATA is applicable. In particular, note the reference to the possible "trigger" property of certain CSS areas.

Reception

A valid NCQ will cause the p-entity to process the CSS area update. Further information under "reception" under PP-DATA is applicable.

PP-FREECONTROL (FCQ)

Purpose

Used to convey CSS information which, due to the defined nature of the relevant CSS area, is not subject to any token control. No response is solicited as part of this protocol message group.

Contents

The parameters below are derived directly from the parameters shown as relevant to the P(VT)-FREECSS request primitive:

p-css-area-ident, p-css-info.

Generation

Generated by p-entity when the p-user issues a P(VT)-FREECSS request primitive. Transmission of a FCQ message will occur as soon as possible, i.e. subject to any necessary token permission at the protocol level. Sequence with other FCQ and non-data messages must be maintained but sequence with NCQ and NDQ messages is unimportant.

Reception

A valid FCQ will cause the p-entity to process the CSS area update and give a P(VT)-FREECSS indication primitive to the p-user.

PP-DELIVER (DLQ)

Purpose

Used to designate delivery points in the stream of NDQ and NCQ protocol messages and, optionally by a parameter, solicit an acknowledgement by an ARQ message.

Contents

The parameters below are derived directly from the parameters shown as relevant to P(VT)-DELIVER request primitive;

pp-token-ident, pp-token-ident-list, pp-solicit-ack.

Some parameters may not be present.

See Additional information.

Generation

Generated by p-entity if p-user issues P(VT)-DELIVER request primitive, following (logically) transmission of any NDQ and/or NCQ protocol messages relating to updates covered by the designated tokens.

Reception

A valid DLQ protocol message will cause the p-entity to give a P(VT)-DELIVER indication primitive to the p-user, with notification of any update information for CCA components covered by the designated tokens not already passed.

Additional information

Presence of the pp-solicit-ack parameter will cause both p-entities to suspend the permission granted by the designated write-access-token(s); see also Additional information to ARQ below.

PP-ACK-RECEIPT (ARQ)

Purpose

Used to acknowledge a reception as requested by a DLQ protocol message. No response is explicitly solicited as part of this protocol structure.

Contents

The parameters below are derived directly from the parameters shown as relevant to P(VT)-ACK-RECEIPT request primitive;

pp-token-ident, pp-token-ident-list.

Some parameters may not be present.

See Additional information.

Generation

Generated by p-entity if p-user issues P(VT)-ACK-RECEIPT request primitive.

Reception

A valid ARQ protocol message will cause the p-entity to give a P(VT)-ACK-RECEIPT indication primitive to the p-user.

Additional information

Both p-entities will release the suspension on the designated write-access-tokens, see PP-DELIVER.

3.3 Protocol Encoding

The encoding of protocol messages conforms to the principles of GPP. GPP defines completely the encoding of those messages which are standardized by it, up to some point at which the parameters become specific-service-dependent, and VTS conforms to this level of standardization.

Where VTS specific messages are defined in this Standard the encoding is defined in this Standard up to a point at which parameters become specific-VTS-Class-dependent.

Parameters which are VTS specific but common to VTS Classes are defined in this Standard up to the point at which sub-structure, if any, becomes class-dependent.

3.3.1 VTS Message Type Codes

In accordance with the rule in GPP for allocation of message type codes, VTS and all VTS Classes use message type codes in the range 64 upwards.

Messages defined for specific VTS Class usage in individual VTS Class standards will use the message type range 128 upwards.

The following table gives the Message Type Codes for the VTS specific protocol messages defined in this Standard.

PROTOCOL MESSAGE		TYPE CODE
PP-GIVE-TOKENS	GTQ	64
PP-REQUEST-TOKENS	RTQ	65
PP-DATA	NDQ	66
PP-CONTROL	NCQ	67
PP-FREECONTROL	FCQ	68
PP-DELIVER	DLQ	70
PP-ACK-RECEIPT	ARQ	71

Values 69 and 72 to 127 are reserved for future standardization.

3.3.2 VTS Parameter Coding Details

The GPP Standard defines a number of "outer level" parameter types and gives some level of definition of the encoding of these parameters. Beyond this it leaves the definition open for use in specific services such as GVT. It does not give a rule or recommendation for the use of parameter type codes for use within compound parameter values in this way or within messages defined specifically in specific service standards such as this one as the interpretation of such parameters is in effect always within the scope of the "outer" parameter or the specific message type.

3.3.2.1 Usage of Parameters Defined in GPP

The table below, continued in the following page, lists the parameters defined in GPP, which are relevant to GVT. Specific encodings, amending or extending those given in GPP, are given in some cases, in others there are forward references to the next level of definition.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
pp-diagnostic	1	-	A	as GPP
pp-transp-data	2	48(1)	T	
pp-protocol-definition	3	-	A	aggregate value, (see 3.3.3.1)
pp-fixed-pe-params	4	-	P	compound value (see 3.3.2.1.1)
pp-initial-pe-params	5	-	P	compound value (see 3.3.2.1.2)
pp-max-pes	6	2	N	as GPP
pp-pe-ident	7	2	N	as GPP
pp-draft-pe	8	2	N	as GPP
pp-dest-encl-type	9	1	S	as GPP

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
pp-pe-specific-params	10 (2)	-	P	compound value, (see 3.3.2.1.3)
pp-save-pe-capab	11	1	S	as GPP
pp-pe-action	15	1	S	as GPP
pp-param-ident-list	16	-	P	compound value, (see 3.3.2.1.4)
pp-param-list	17	-	P	compound value, (see 3.3.2.1.4)
pp-start-pe-option	20	1	S	as GPP
pp-save-pe-option	21	1	S	as GPP
pp-special-conventions	22	2	T	see 3.3.2.1.5
pp-special-info	23	-	T	see 3.3.2.1.5

Additional information

1. This maximum length may be explicitly reduced when this parameter is used in particular protocol messages; the maxima applicable are given in the service element descriptions to which the protocol message definitions refer.

2. In addition to this Standard compound parameter type 10, parameter type codes 64 to 127 are reserved for specific forms of pp-pe-specific-params which are expected to be terminal parameters but may have aggregate values.

3.3.2.1.1 Use and Content of pp-fixed-pe-params

This is VTS class specific. Its content can include pp-protocol-definition, which must then be the first component parameter; see 3.3.3.1. Other possible content is pp-special-conventions, see GPP; if these are to be "fixed". Some classes may make no use of pp-fixed-pe-params.

3.3.2.1.2 Use and Content of pp-initial-pe-params

This is VTS class specific. Its content can include pp-protocol-definition, which must then be the first component parameter; see 3.3.3.1 which

also gives constraints on the value of this when occurring in this way. Other possible content is pp-special-conventions, see GPP, if these are not to be "fixed".

3.3.2.1.3 Content of pp-pe-specific-params

This can include component parameters as given in 3.3.2.1.2 for pp-initial-pe-params, which will then be the first component parameter(s). It can also contain further class-specific parameters.

3.3.2.1.4 Content of pp-param-ident-list and pp-param-list

The definition of the PE parameters from the point in the Directed Graph determined by pp-protocol-definition, and hence the content of parameters pp-param-ident-list and pp-param-list, is class-specific.

3.3.2.1.5 Use and Content of pp-special-conventions and pp-special-info

Availability of these parameters is class dependent. The content is not defined in this Standard and will generally be user-defined.

3.3.2.2 VTS Specific Parameter Definitions

The following table lists the parameters which are standardized in this Standard and conform to the generic principles for parameter encoding. Some special cases are covered in 3.3.3.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
pp-token-ident-list	63	2	M	f1:p-terminate-t'n f2: enclosure-t'n f3: negotiate-t'n f4: transfer-token Other bits reserved for class-specific use.
pp-token-ident	62	1	S	class-specific
pp-encl-service-specific	61	-	class-specific	
pp-solicit-ack	60	1	M	f1=1 "yes" =0 "no"
pp-css-area-ident	59	-	C	user-defined
pp-css-info	58	-	according to css def'n	

3.3.3 Special Parameter Encodings

This sub-clause gives VTS specific detail of special parameters defined in GPP and gives additional special message and parameter encodings defined by GVT.

3.3.3.1 Encoding of Parameter pp-protocol-definition to PP-CONNECT-REQ (CNQ) and PP-CONNECT-RESP (CNR)

The value of this parameter is an aggregate consisting of the following fields, conforming to GPP:

pp-service-ident: Single octet, S type value:
2: "VTS" is applicable for GVT, all classes.
0: field is "void"; see Additional information.

pp-service/class-version-idents: Single octet, M type value:
version applies to the VTS class; applicable values are defined in class standards.
All bits 0: field is "void"; see Additional info.
(bit 8 reserved as in GPP and set to 0)

pp-service/class-subset-ident: single octet, M type value:
subset, encoded on bits f1 through f4, applies to the VTS class; applicable values are defined in class standards.
Bit f7: = 1 field value is "void", other bits set to 0; see Additional information.
Bit f6: = 1 "delivery-control", = "no del-control",
Bit f5: reserved for future standardization,
Bit f8: reserved and set to 0; see GPP.

pp-service-class-ident: Single octet, S type value:
1 : "VTS basic class".
2 : "VTS forms class".
Other values reserved for further VTS classes.
0 : field is "void"; see Additional information.
This field is omitted (octet not present in parameter value) if it is "void" and the pp-pe-profile field is not required.

pp-pe-profile: Variable length, A type value:
use and encoding of this field is class specific and is defined in class standards. This field is omitted if not required.

Additional information

The three uses of pp-protocol-definition given in GPP, are relevant to this version of GVT: dependent VTS classes may not use all of them as defined in individual class standards.

With this version of GVT further rules for use of the parameter and its fields are as follows:

a) When encoded at the outer level of parameter encoding in CNQ and CNR messages, (and thus having "fixed" status), the fields pp-service-ident, pp-service/class-version-idents and pp-service-class-ident are mandatory with the "non-void" values defined above (since with this version of GVT and GPP these parameters of the p-connection must be "fixed"). The other fields defined above are optional. Sub-clause 3.3.2.1.1 gives further information.

b) When encoded as the first component of pp-fixed-pe-params (in CNQ and CNR messages) the same rule applies as case (a) above. Sub-clause 3.3.2.1.1 gives further information.

With this version of GVT, pp-protocol-definition must occur exactly once in one or other of the above ways in each of CNQ and CNR; dependent class standards may choose to apply further restrictions.

c) When encoded as the first parameter of pp-initial-pe-params or pp-pe-specific-params the fields given in (a) as mandatory must have "void" values. The other fields defined above can take values as defined above. Sub-clauses 3.3.2.1.2 and 3.3.2.1.3 give further information.

3.4 Session Service Mapping

VTS conforms, in general, to the statements on Session Service mapping in Clause 3.4 of GPP. Additional information is needed on the derivation of session parameters from the VTS service and protocol characteristics, and also on the usage of certain session service facilities which are not covered in GPP because the usage is specific-service-dependent.

3.4.1 Session Connection Establishment

Additional information is needed on the derivation of the parameters pp-session-subset and pp-session-params. Currently this will be specified in individual VTS Class Standards; general statements may be added to this Standard in a later version. The individual class information will include the session subset(s) used or available (according to the service parameters requested by the p-user); this will imply which session tokens are available, and may enable the initial owner of certain tokens to be agreed.

3.4.2 Session Connection Termination

VTS will use the session terminate-token to mediate the issue of S-RELEASE request primitive where the applicable session subset provides this token (i.e. the p-terminate-token is mapped onto the s-terminate-token). Where the s-terminate-token is not available the p-terminate-token is achieved in some appropriate manner; if, for example, it is mapped onto the s-data-token it then cannot be owned separately from any other p-tokens mapped onto the s-data-token. Individual class standards will define the applicable case(s).

3.4.3 Session Dialogue Controls

The use of these session facilities is VTS-Class-dependent. It is likely (and may be stated definitively in a later version of this Standard) that either the session data-token or the session major synchronization facility, and possibly both, will be used.

Individual VTS Class Standards will define usage of any other session facilities not covered by GPP or GVT and the mapping of the class-dependent protocol messages into the appropriate session service primitives. Such usage of session facilities may not be directly visible at the p-service interface.

A later version of GVT may include VTS standard usage of some other session service facilities.

3.4.4 Session Data Transfer Facility

Protocol Messages not covered by the above cases are all mapped into the SSDU parameter of the S-DATA primitive.

Appendices

APPENDIX A

BRIEF DESCRIPTION OF THE REFERENCE MODEL FOR OPEN SYSTEMS INTERCONNECTION

A.1 Scope

This appendix 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 Figure A1):

a) application processes within an OSI environment

b) connections which permit information exchange,

c) the systems themselves.

Note

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 comprehensive description of the Reference Model.

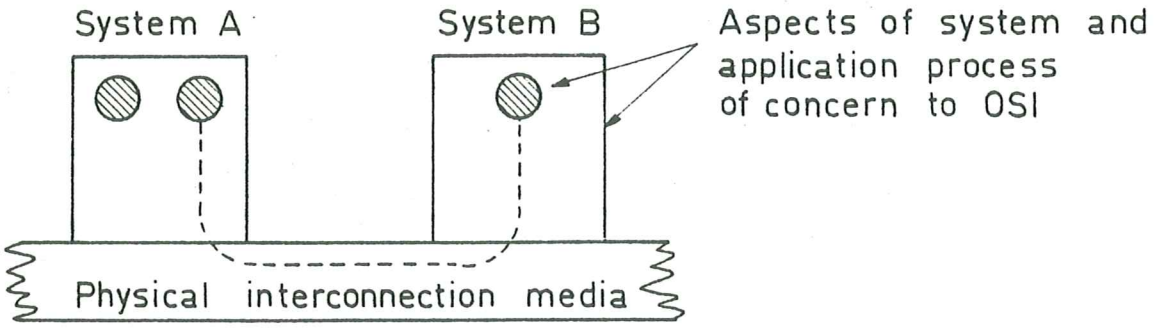


Fig. A1 - General Schematic Diagram Illustrating the Basic Elements of Open Systems Interconnection

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. A2.

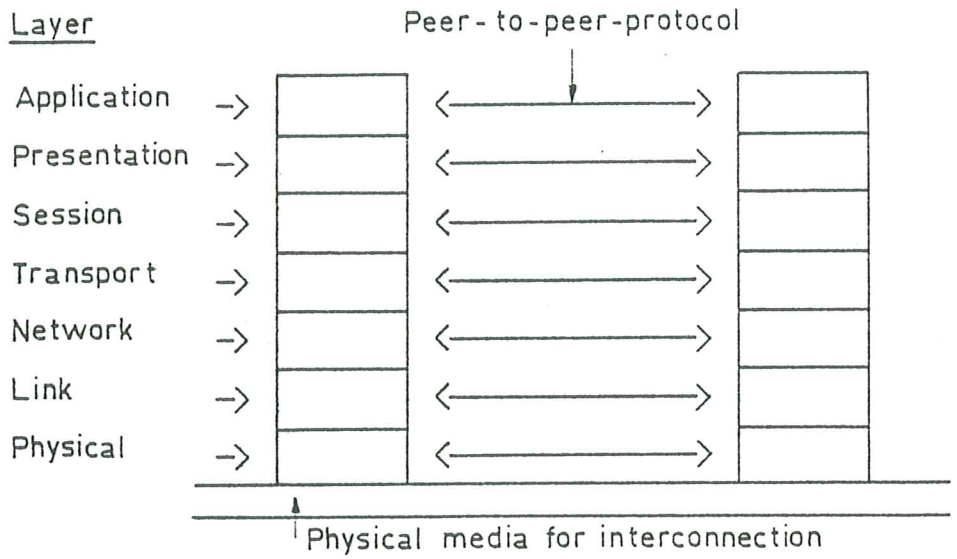


Fig. A2 - The Seven-Layer Reference Model and Peer-to-peer Protocol

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 principles 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. The 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 procedural means to exchange network service data units between two transport entities over network connections.

A.3.6 The Data Link Layer

The purpose of the Data Link Layer is to provide the functional and procedural means to activate, maintain and deactivate one or more data link connections 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 Data 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 with the Physical Layer.

APPENDIX B

INDEX AND GLOSSARY OF TERMS

B.1 General

The terminology used in this Standard consists of:

- Terminology defined in the Reference Model for Open Systems Interconnection.
- Terminology for concepts of Presentation Layer in general and for Virtual Terminal Service and Protocol in particular, which is defined in this Appendix, see B.2.
- Notation terminology, which is defined in Appendix C.

An index of accronyms is provided in B.3.

B.2 Glossary of Terms

For the purpose of this Standard the following terms have the meaning indicated. A reference is given to the relevant clauses of the Standard.

Access Control, AC (2.2.4, 2.3.7.3)

The component of the CCA which controls the access (for inspection or update) by the p-users to the information in the CDS or CSS.

Active Location (2.3.7.5)

The currently addressed cell (storage position) in the CDS.

Application Program (2.1)

The user of the Virtual Terminal Services within a system.

Atomic Object (2.2.2)

The smallest separately addressable unit of information which has a distinct semantic significance within the contest of the Virtual Terminal Service. This unit may itself consist of a number of parts with differing types of semantic significance: i.e. the Primary Value and Attribute Values.

Attribute (2.2.3)

A property associated with an item or a group of items of information in the CDS or other component of the CCA.

Cell (2.2.2, 2.3.7)

The smallest individually addressable storage element of the CDS; it has a single, unique value of the Pointer and can contain one Atomic Object.

Class (2.1.1)

Subset of total Virtual Terminal Service of Presentation Layer corresponding to a limited number of functions needed for a type of use of the service; corresponding to this there is a subset of the total protocol messages and sequences of messages.

Conceptual Communication Area, CCA (2.1)

This is the conceptual repository for all the information, both data and control, which is shared between the two p-users for the purpose of their Virtual Terminal service usage. Within the scope of VTS there is no other means of communication between the p-users.

Conceptual Image (2.1)

The information content of the CDS as viewed by the p-users, i.e. interpreted with reference to the DSD, Access Control and Attribute values.

Conceptual Data Store, CDS (2.1, 2.2.2)

The principal (conceptual) store for data information shared between two presentation users using VTS. It is part of the CCA.

CSS Area (2.1, 2.2.5)

One of possibly several areas in the CCA, defined by default or explicitly by the p-user during negotiation, used for exchange of (device) control, signalling or status information between the p-users. Used to support a conceptual CSS mechanism.

CSS Mechanism (2.2.6)

A conceptual control, signalling or status mechanism which the p-users visualize as capable of transferring (device) control, signalling or status information. Each is achieved by a CSS area in the CCA associated with (one or more) virtual devices and with CSS mechanism realization parameters. Such areas are defined or defaulted and can be used to provide the p-users with (logical) device control, alarms, interrupts, light pens, function keys, status information and so on.

Current PE (2.3.7)

The Defined PE which is in use in transfer-enclosure.

Data Structure Definition, DSD (2.1, 2.2.1)

The part of the CCA which holds control information including the agreed service parameters of any Defined PEs. This includes the mutually agreed structure of the CDS. The DSD and AC together allow the interpretation of the data in the CDS as a conceptual image available to the p-users.

Defined PE (2.3.6)

A PE which is in a completely agreed state such that it forms one particular usable set of presentation parameters.

Destructive (in the context of effects of services)

See Appendix C.

Dimension (2.2.1, 2.3.7)

A term used, not necessarily in precisely the same way for all classes, to refer to one axis of a CDS whose contents are addressed by means of a coordinate system (discrete or continuous). Each dimension may or may not be bounded.

Directed Graph for Parameters (2.3.6.4)

The method used for representing the relationships between presentation parameters; such relationships constrain the process of negotiation of a PE.

Disestablishment Phase (2.3.3.2)

The transient phase, consisting of a single service element, whereby a presentation-connection is terminated.

Draft PE (2.3.6)

The PE being created or amended within negotiation-enclosure during a multiple-interaction negotiation.

Enclosure (2.3.5)

A phase of the p-connection in which activities of a particular or possibly restricted type are performed. All interaction activities initiated within an enclosure are completed before proceeding to the next phase and thus no activities are allowed to "penetrate" into following enclosures. This does not prevent the information resulting from activity in one enclosure being used in a following enclosure.

Enclosure-token (2.3.5)

A token which is used (in VTS) to mediate the issue of requests for a change of enclosure.

Establishment Phase (2.3.3.1)

The transient phase, consisting of a single service element, whereby a presentation-connection is set up between a pair of peer p-users.

Multiple-interaction Negotiation (2.3.6.2)

The type of negotiation used within a negotiation-enclosure to create or modify a Defined PE. Only one PE can be negotiated by one occurrence of negotiation-enclosure. During the operation presentation parameters can be negotiated individually or in groups. A PE under negotiation does not become a new Defined

PE or replace an existing Defined PE unless and until the negotiation-enclosure is successfully terminated.

Multiple PE Option (2.1, 2.3.7)

The optional capability for simultaneously holding more than one Defined PE definition during a p-connection, although at most one can actually be in use (be the Current PE) at any instant.

Negotiate-token (2.3.6.2)

A token used to mediate which p-user can issue service requests during multiple-interaction negotiation.

Negotiation (2.3.6)

The action of redefining an existing PE or creating a new PE. The ability to create new PEs may be restricted by establishment time options (multiple PE option).

Negotiation-enclosure (2.3.6.2)

The phase in a p-connection where the p-users may negotiate using multiple-interaction negotiation. The ability to use this type of enclosure is subject to agreement between the two p-users.

Negotiation Group (2.3.6.3)

A set of related parameters currently being negotiated.

Null-enclosure (2.3.2)

The type of enclosure entered after establishment of a p-connection if there is not a Defined PE, or at other times when the users so desire; there is no Current PE in null-enclosure. The ability to use this type of enclosure is subject to agreement between the two p-users.

Parameter (in the context of negotiation) (2.3.6.3, 2.3.6.4)

The objects of a negotiation operation whose values subsequently determine the precise nature of the p-service available on the p-connection. Parameters can be grouped hierarchically to form sets of related parameters using the directed graph concept.

Pointer (2.3.7.5)

A pointer designates the currently addressed cell (storage element) in the CDS. Its form depends on the structure of the CDS in a particular case; typically it could consist of 1, 2 or 3 coordinates depending on the dimensionality of a CDS.

Presentation Environment, PE (2.1, 2.2.1)

A complete and self-consistent set of values for the parameters of the p-service. It includes any parameter specific to a particular p-service operating within the framework of the GPP.

It is not necessary for all possible presentation parameters to have values to complete a particular PE. Depending on establishment time options one or more PEs can be in defined state (Defined PE) at the same time on one p-connection. At most one PE can be in use (Current PE). All PEs cease to exist when the p-connection is disestablished.

Protocol Message (3.1.2)

The smallest unit of information transferred across the Open Systems Interconnection having semantic significance to the VTP p-entity.

Real Device

Real device designates a specific hardware device.

Rendition

A style of visually displaying characters.

Root Parameter (2.3.6.4)

The highest order parameter of a negotiation group, i.e. nearest the head of the directed graph.

Sequenced (in the context of effects of services)

See Appendix C.

Service Element (2.1)

The smallest unit of activity of the Service with self-contained semantic significance which can be initiated by one service-user and which may or may not have a consequential effect on another service-user. A service element consists of one or more Service Primitives issued by or received by one or both service-users. (See also Appendix C.)

Service Primitive (2.1)

The smallest unit of activity on the conceptual service interface at one service access point; one or more service primitives at one or both SAPs make up a Service Element. Note that there is neither a one-to-one nor indeed any specific mapping implied between these service primitives and protocol structures or protocol messages. Only primitives of the conceptual service interface are referred to in this Standard. Rules for defining real service primitives and for use of macros as part of a real service interface are not the concern of this Standard. (See also Appendix C.)

Single-interaction Negotiation (2.3.6.1)

The negotiation consisting of only a single confirmed service element, available in either null-enclosure or transfer-enclosure; no change of enclosure can be effected by it.

Token (2.2.4, 2.3.4)

An abstract object with which is associated some capability available to a pair of peer entities, and which mediates the

actual use of the capability to avoid collision situations. Any token is owned at any instant by at most one of the entities. A token of the Presentation Layer may be a service-token visible to the p-users and mediating their activities, or a protocol-token used by the p-entities to mediate protocol activities and not directly visible to the p-users.

Transfer-enclosure (2.1, 2.3.7)

The type of enclosure in which the p-users may operate on data and control information (as distinct from parameters, see negotiation-enclosure). This will be performed within the context of a selected Defined PE, the Current PE.

Virtual Terminal Protocol, VTP (2.1)

The set of Protocol Messages and the rules for their use which enable the Virtual Terminal Service to be supported. The total VTP is divided into classes which are agreed subsets of the total and defined in class specifications.

Virtual Devices (2.2.6, 2.3.9)

One or more conceptual devices in the virtual terminal on which the conceptual image defined by the data in the CDS and DSD may be visualized.

Virtual Terminal Service, VTS (2.1)

One of the major categories of service offered by OSI.

Virtual Terminal (2.1)

One view of this is just as the CCA and the data it contains. An alternative view is as a device or group of devices, including control, signalling and status mechanisms where appropriate, on which the two p-users visualize the data of the CCA appropriately interpreted. These views are not incompatible, differing only in the degree of abstraction.

B.3 Index of Acronyms

AC	Access Control
BNF	Backus-Naur Format
CCA	Conceptual Communications Area
CDS	Conceptual Data Store
CICI	Conceptual Interface Command Interpreter
CSS	Control, Signalling and Status (applied to area or mechanism)
DSD	Data Structure Definition
GPP	Generic Presentation Protocol
GPS	Generic Presentation Service

GVT	Generic Virtual Terminal
II	Indication-Indication (applied to service structure)
PE	Presentation Environment
PPDU	Presentation Protocol Data Unit
PSAP	Presentation Service Access Point
RC	Request-Confirmation (applied to service structure)
RI	Request-Indication (applied to service structure)
RO	Request Only (applied to service structure)
SSDU	Session Service Data Unit
TLV	Type, Length, Value (applied to encoding technique)
TWA	Two-Way Alternate (as a dialogue discipline)
TWS	Two-Way Simultaneous (as a dialogue discipline)
VTP	Virtual Terminal Protocol
VTS	Virtual Terminal Service

APPENDIX C

SERVICE DESCRIPTION TECHNIQUE

The terminology and description technique used in this Standard is in accord with Standard ECMA-86.

The Service Description is in terms of a number of "Service Elements". A Service Element is an elementary operation of the Service viewed as a whole, i.e. taking account of both the Service Access Points at which the service-users interact with the service. A Service Element thus consists of one or more events on the conceptual service interface at one or both service access points. These events are known as service primitives, frequently abbreviated to just primitive. Primitives, depending on the defined structure of the service element, can occur in one or more of the event forms "request" "indication" "response" "confirmation" (see Standard ECMA-86). Parameters will usually be associated with the primitives, and may be supplied by the issuing layer-entity or "filled-in" by the receiving layer entity for the use of the issuer.

The four types of service element defined in Standard ECMA-86 (known as service structures) are given mnemonic names in this Standard as follows:

- RI Type 1, i.e. Request-Indication
- RC Type 2, i.e. Request-Indication-Response-Confirmation
- II Type 3, i.e. Indication-Indication
- RO Type 4, i.e. Request-Only.

For each of the service elements relevant to a particular part of the total presentation-service the following information is given in the sub-clauses of the Service Description:

- the name of the primitive (as title),
- the purpose of the primitive,
- the type(s) of service elements (and hence the applicable event forms in which the service element can occur),
- a table showing the applicable forms of the primitives and the parameters which can accompany the primitive for each applicable form, with the entries for each listed parameter conforming to the following key:
  - D : the parameter is supplied by the user,
  - U : the parameter is supplied by the service,

  - D,U: the U form is the combination of the user-supplied parameter (D) and the service-supplied parameter,
  - x : the parameter is not used in the particular form of the primitive,
  - (n) : indicates that the parameter can be multi-valued,
  - (1) : indicates that the parameter must be single-valued (usually shown in response to a multi-valued request parameter). (1) is the default if no entry is made.
- notes on the use of the primitive and/or parameters,
- effects of the primitive and effects, if any, on other primitives, issued before and after the event(s) of this primitive,
- additional information as necessary.

Where effects are described as sequenced this means, as a general rule, that with respect to other services described as sequenced, indications appear in the same sequence as requests, and confirmations appear in the same sequence as responses (where the service structures contain these elements). The protocol definition will determine whether or not response/confirmation pairs are always in the same sequence as their associated request/indication pairs - this is not necessarily implied by the term sequenced. Any relative sequencing not covered by this general rule will be specified explicitly with the service element description.

Where effects are described as destructive this means that events (including data delivery) associated with previously initiated services may not occur.