ECMA
EUROPEAN COMPUTER MANUFACTURERS ASSOCIATION

STANDARD ECMA-88

BASIC CLASS VIRTUAL TERMINAL

SERVICE DESCRIPTION AND PROTOCOL DEFINITION

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-88

BASIC CLASS VIRTUAL TERMINAL

SERVICE DESCRIPTION AND PROTOCOL DEFINITION

March 1983

BRIEF HISTORY

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

This Standard ECMA-88 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-88 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.

This Standard is within the framework for the co-ordination 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 oriented towards urgent and well understood needs. It is intended to be capable of modular extension to cover future developments in technology and needs.

The Basic Class of Virtual Terminal Service

The ECMA Generic Virtual Terminal Service and Protocol Description (GVT), ECMA-87, introduces the concept of Virtual Terminal Service (VTS) and shows how this relates to the Generic Presentation Services (GPS), ECMA-86, of the Presentation Layer of the ISO Reference Model of Open Systems Interconnection. This Standard ECMA-88 introduces the concept of Classes of Service of the VTS. The purpose of this Standard is to describe the service for the Basic Class Virtual Terminal and to define the Basic Class Virtual Terminal Protocol (VTP). This Standard assumes that the reader is familiar with the GVT. It makes frequent reference to the concepts and terms of that Standard ECMA-87, which through these references becomes an implicit part of this Standard. ECMA-87 itself makes reference to ECMA-86 and these references are assumed by this Standard for the service description. The protocol definition, however, is complete and free-standing, taking into this Standard the relevant parts of the GPP and GVT standards.

The Basic Class VTS is primarily expected to be used for communication between an application program presentation-service-user (p-user) in a "computer system" and a human

(operator) p-user at a "terminal" supported by another "computer system", the two systems being interconnected by means of OSI. The Basic Class Virtual Terminal Protocol defined by this Standard is for OSI exchange between two systems providing a VTS Basic Class OSI service. Because the service is described and the protocol defined in a symmetric way in this Standard, the service can be used by systems supporting a pair of human p-users or a pair of application p-users if the Standard meets the requirements of such p-users.

TABLE OF CONTENTS

Table of Contents (cont'd)

Table of Contents (cont'd)

1 General

1. GENERAL

1.1 Scope

This Standard ECMA-88 provides specifically for the Basic Class of the Virtual Terminal Service of the Presentation Layer of the Reference Model of Open Systems Interconnection the following:

a) the description of the Virtual Terminal Model Components,

b) the description of the Services and Functional Features of the Virtual Terminal Service,

as the particularization for Basic Class of the general statements of the concepts of Virtual Terminal Services and Protocols in Open Systems Interconnection given in ECMA-87, (GVT)

c) the definition of the Virtual Terminal Protocol (VTP) Messages and their Encodings,

d) the definition of the Protocol Rules for the construction of valid sequences of protocol elements,

e) the definition of the usage made of services of Session Layer, and the mapping of Presentation Protocol Elements into the Session Service Primitives,

f) the statement of the Conformance Requirements for implementation of Basic Class VTP according to this Standard.

This Standard thus defines the interactions between two system environments supporting two presentation-serivce-users (p-users) via the Basic Class of Virtual Terminal Service.

This Standard ECMA-88 does not define any other interactions between a p-user and the presentation service (p-service).

This Standard ECMA-88 is not an implementation specification for information processing systems.

1.2 Field of Application

This Standard ECMA-88 is provided for use by:

a) implementors of equipment required to provide functionality at presentation level to support Basic Class of Virtual Terminal Service for the purposes of Open Systems Interconnection,

b) implementors of equipment required to provide functionality at application level making use of Basic Class of Virtual Terminal Service for the purposes of Open Systems Interconnection,

c) potential presentation-service-users interacting with equipment as in b) by means of Basic Class of Virtual Terminal Service.

Basic Class is formally applicable to unstructured sequential, line-orientated or page-orientated modes of presentation on alphanumeric devices, and this Standard is written in terms applicable to such usage in particular. This does not, however, preclude the provision by either p-user of local conversion functions to achieve other forms of usage, but such functions are wholly outside the scope of this Standard.

With reference to the classification scheme in the ECMA Generic Virtual Terminal Service and Protocol Description (GVT) provided in ECMA-87, the Basic Class is defined as follows:

- Atomic Object: Character
- CDS Structure: Contiguous cells, in a single 1, 2 or 3-dimensional uniform array.

For Basic Class a two-way-alternate (TWA) update (write) access control is imposed on the whole CDS.

1.3 References

ECMA-6 : 7-Bit Input/Output Coded Character Set

ECMA-35 : Code Extension Techniques

ECMA-43 : 8-Bit Coded Character Set

ECMA-48 : Additional Control Functions for Character-Imaging I/O Devices

ECMA-75 : Session Protocol

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

ECMA-87 : Generic Virtual Terminal - Service and Protocol Description

ISO 6937 : Coded Character Sets for Text Communication

ISO 7498 : Reference Model of Open Systems Interconnection

2 Service

2. SERVICE

2.1 The Generic Model

The definition of the Virtual Terminal Service and Protocol is based on the concept of a Virtual Terminal, as fully described in the GVT, ECMA-87, which also establishes the relationship of VTS to the Generic Presentation Services (GPS) of Standard ECMA-86 and Basic Class is a particular case conforming to this principle. The following material is an abstract from this intended to aid the clarity of this Standard but does not render a study of ECMA-87 unnecessary.

The Model of a Virtual Terminal is described in a Conceptual Communication Area, which is a (conceptual) repository for all information being exchanged between or of mutual interest to the p-users. The CCA has normally no physical existence, each p-user will have its own realization.

Using information stored in the CCA each p-user is able to derive a visualization of the Virtual Device(s). 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-user is able to control, to signal to, and to obtain status and signals from, the Virtual Devices by use of the CSS (Control, Signalling and Status) areas of the CCA.

For the purpose of the Basic Class the Conceptual Communication Area contains the following information structures:

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

The following diagram illustrates this Model. Explanatory notes on the operation of the Model, and in particular the important relationship between the common conceptual service interface and the two separate real interfaces are given in the GVT.

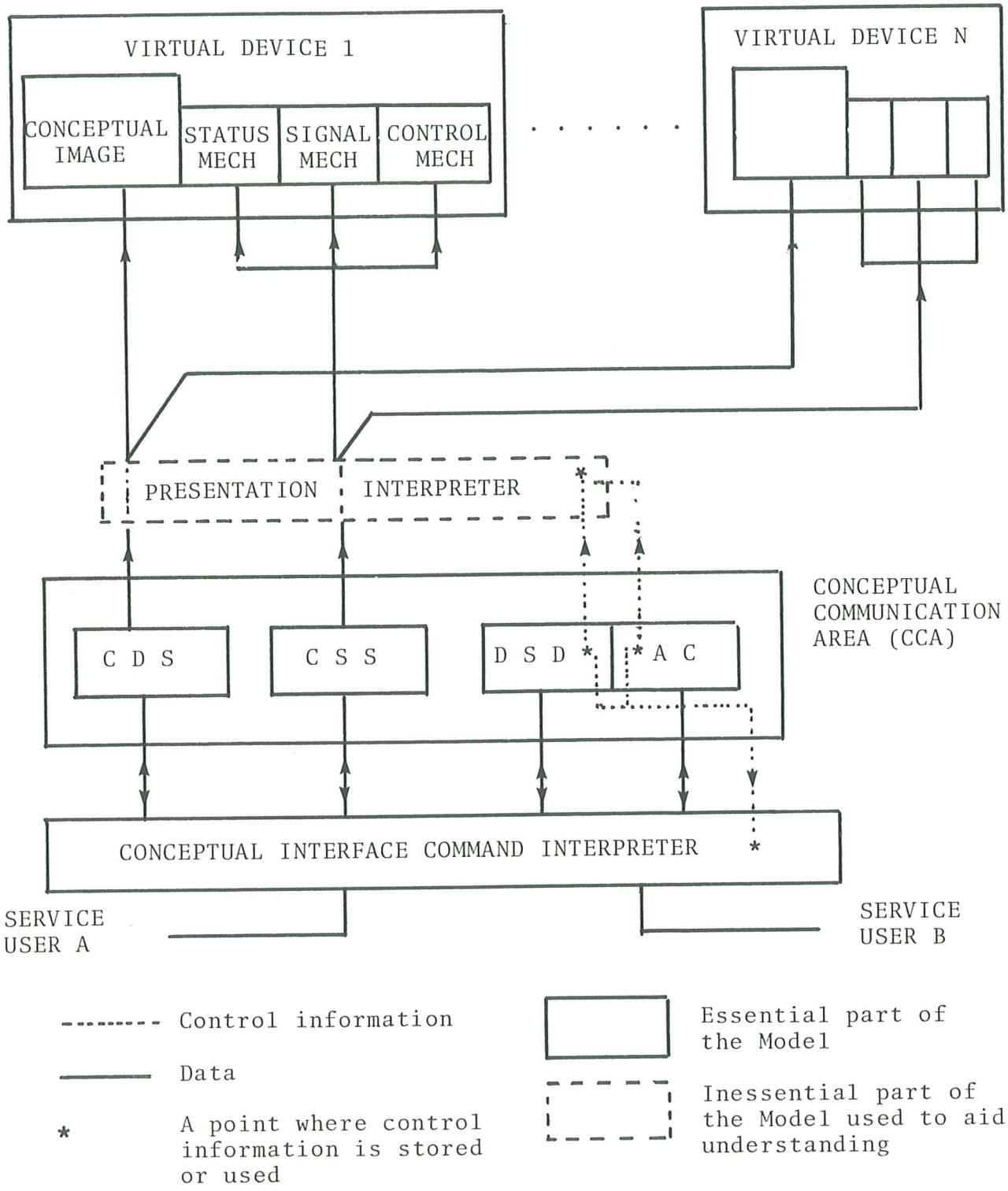


Figure 1. The Model View (Conceptualization of Service)

The above diagram 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 (subject to having write access control) by means of conceptual service interface requests and can selectively make such information accessible. 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 it is in fact "delivered" to the peer p-user by use of the access control facilities (delivery control and give-token).

Whether a p-user has the right to access the CDS is controlled by the access control information in the AC. Service primitives are provided so that the p-users can change the state of the access control information. For example, a p-user can relinquish the exclusive write access or request the exclusive write access.

The diagram also shows the conceptual visualization of the information in the CDS on one or more Virtual Devices, 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 signaling 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.

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

2.2 Service Overview

This clause describes the CCA available in Basic Class of VTS including the single Presentation Environment (PE) used in Basic Class and the facilities for defining this PE and changing it. The components of the CCA conform to the principles of GVT. This clause describes these components in detail. Clause 2.3 gives the service facilities which enable these components to be used.

2.2.1 Particularization of Components of the Model for Basic Class

The functions of these components are described in GVT. This clause gives an introduction to their particular nature in the Basic Class, this relating at least in part to the classification scheme given in GVT. Later subclauses expand on this general information.

This version of Basic Class provides for a maximum of one defined Presentation Environment (PE) at any one time (but provides optional facilities enabling it to be changed). Thus the Multiple-PE option as in ECMA-86 and ECMA-87 is not available.

The semantically indivisible (atomic) objects in Basic Class are single characters. These will often be the well known sets of alphanumerical characters, punctuation marks and other symbols, but the use of mosaic characters, or of user-defined symbols, is possible. The use of composite characters (e.g. accented letters) in the range of values for the atomic object is not excluded. Facilities are provided for designating the standard set(s) of characters applicable, and how these are to be coded. Each atomic object is discrete; the p-service is not concerned with any semantic relationship between atomic objects, either due to their contiguity in the addressing structure (see below) or due to any other inherent property known to the p-users.

One atomic object, however complex its definition, is wholly contained in one cell of the CDS; one cell holds at most one complete atomic object. For example, an accented character could be contained in one cell if the atomic object is so defined.

The structuring information in the DSD for the Basic Class describes how the cells are arranged into an ordered uniform array of one, two or three dimensions each of which may be bounded or unbounded. This provides great flexibility in local mapping to real devices. Such structuring information is placed in the DSD during negotiation for a basic class presentation service connection. The purpose of the array is only to enable addressing of the cells for the purpose of placing or accessing atomic objects in them. Basic Class VTS does not define any other explicit semantic significance to the array structure. In particular, it does not provide explicit control over the size of objects when realized on a medium, or over inter-object gap or overlap in any dimension.

Basic Class does not provide explicitly for the association of explicit semantic significance to a sub-group of cells other than as a result of the above dimensionality, or for the attachment of an explicit property (attribute) to any sub-group of cells; by sub-group is meant some collection less than the total CDS. Some attributes are applicable to the total CDS/CCA.

Basic Class provides default and p-user negotiated CSS areas for control of, for signalling to and from, and for status information from, the Virtual Device as described in

GVT. Each CSS area is either subject to access control or (totally) uncontrolled; in general this is subject to individual negotiation.

In Basic Class write-access to the CDS, and to all CSS areas defined as being access-controlled, is on a strict two-way-alternate basis. All such access control is by means of a single write-access-token, possession of which can be passed between the two p-users. No separate explicit Read-Access Control is provided in Basic Class. Read-Access to the CCA to examine the content always accompanies write-access, but the p-users may negotiate the delivery control facility in which case the p-user having write-access can control the points in the update of the CDS at which the updates are made available to the other p-user. The reliance on information released in this way is a matter of convention between the p-users. The Basic Class standard offers no facilities for enforcing such conventions, and similarly provides no facilities for enforcing any p-user conventions for use of uncontrolled CSS areas.

Basic Class allows a large range of capability and facilities to be handled within its framework and the definition of a particular Virtual Terminal instance is embodied in the set of Parameters and their Values set up at the time of establishment of the presentation-connection (and possibly varied by re-negotiation during the life of the presentation-connection) and all interactions by the two presentation-users must be within the domain of the current set of parameter values, i.e. the Presentation Environment (PE). The applicable parameters for Basic Class are shown in a directed graph in sub-clause 2.3.6.3 where each is given a reference number which is used in the remainder of the text as appropriate.

The following sub-clauses of clause 2.2 give further detail and definition of the various possible components of a Virtual Terminal of Basic Class.

2.2.2 Basic Class Atomic Objects

A single type of Atomic Object is defined in Basic Class and has two components:

a) Primary Value: In Basic Class this represents (i.e. is some encoding for) a particular "character" from the character set,

b) Attribute Values: In Basic Class the Object Attributes are Character Attributes, i.e. apply to a single character.

An atomic object in the CDS represents a discrete element of p-user information, wholly contained, conceptually, within the (single) cell of the CDS in which it has been placed. The Basic Class p-service is not concerned with the interpretation placed on the atomic objects themselves by the p-users or on any relationship across the boundary between contiguous cells of the addressing structure assumed by the p-users other than the ordering implied by the addressing structure.

The Values given to the various Character Attributes determine the meaning of the Primary Value and can also give various secondary characteristics to the object, for example determine the style of visual imaging of the object on a screen or printer.

For each such attribute, use of which has been agreed as part of the PE, there will be a valid range of permissible values and a default value which will apply initially to all Cells of the CDS. Further definition of attributes is given in sub-clause 2.2.4.

There is a special value of Primary Value called "Nil" which exists before a cell is written and can be restored by an erase action. A cell with Primary Value "Nil" is deemed to be empty; although there may still be attribute values within the cell, there is nothing to which they can apply (to allow "visibility" of background attributes, "space" must be written as Primary Value).

2.2.3 Data Structure Definition for Basic Class

The DSD holds the parameters which define all aspects of the CCA including the way in which the cells of the CDS are organized for the purposes of access, addressing and control. It holds the parameters of the atomic object definition, in particular the character attributes which are to be applicable and for each one, the range of values and the default value; certain other attributes held in the DSD refer to the whole CCA or CDS (are "global"). It holds the parameters relating to the "realization" of the CDS information on virtual devices. It holds the parameters describing the CSS areas associated with the virtual devices. This collection of information is the main part of the presentation environment for the Virtual Terminal.

For Basic Class of VTS, the organization of the cells of the CDS is defined in terms of a number of "Dimensions". The concept of dimensions and the definition of the dimensions for Basic Class are covered by the following subclauses. All the CDS cells are identical in their ability to hold, conceptually, one and only one complete atomic object and all relevant character attributes.

2.2.3.1 General Concepts of Dimensions and Addressing

The dimensions used in this Standard to describe the data structure applicable to Basic Class form a dimension hierarchy. Each dimension is considered to be an ordered set of the elements applicable to that dimension.

A particular point on a dimension is designated by one integer value of the coordinate for that dimension, and identifies implicitly and uniquely one of these elements.

The ordering implies that the element designated implicitly by one particular value N of the coordinate of the dimension is deemed to be logically after the element designated implicitly by value N-1 of this coordinate and logically before the element designated by value N+1. Thus the elements associated with a dimension can be addressed for use in sequence by a sequential use of the numerical values 1,2,.., N,N+1,N+2,... of the coordiante for the dimension. Value 1 is termed the origin of the dimension.

The significance of the dimension hierarchy is that the elements for a particular dimension are each composed of some structure of (array of elements on) the lower order dimensions.

The points on any dimension are always dense, there cannot be "missing" values; thus conceptually from the dimension aspect all the elements exist and can be addressed.

The elements on the lowest-order dimension are deemed to be atomic in that the dimension structure cannot be used to sub-divide these elements in any way and each one can only be addressed as a whole. Each such atomic element is known as a cell and is deemed to be capable of holding a single atomic object. The dimension aspect is not concerned with the nature of the atomic objects; sub-clause 2.2.2 defines the atomic object for Basic Class.

A cell which exists but has not been explicitly "written" will contain primary value "nil" and default values for the applicable character attributes. The generation of cells is covered in following sub-clauses.

2.2.3.1.1 Bounded and Unbounded Dimensions

There is no theoretical limit to the value of a coordinate, but a particular dimension will be defined as either bounded or unbounded when its use is agreed.

If a dimension is bounded this means that there is a maximum value of coordinate which can usefully/ legitimately be used for this dimension. This maximum value is known as the bound of the dimension.

The notation Yb is used to denote the value of bound for Y dimension, and similarly for other dimensions.

An array on a bounded dimension can only exist as a whole, i.e. if it exists at all, due to a coordinate value on the higher-order dimension existing, all its elements exist; the minimum form in which these can exist depends on the form of these elements (they may be atomic cells, or arrays, bounded or unbounded, on a lower-order dimension).

The permissible values of coordinate to address the elements of a bounded dimension are 1,2,3,....,Yb, and no others, taking Y as an example. There is, however, one special case, for the lowest-order dimension (X), in which a coordinate value greater by 1 than this maximum is deemed to be possible; this is known as limit and designated as XL, i.e. XL = Xb+1; there can be no cell or object at limit.

With an unbounded dimension there is no bound and the coordinate can take any (positive) value. At any particular time all and only the points on the dimension corresponding to coordinate values equal to, and lower than, the maximum value which has existed are deemed to exist and to be addressable. A cell (or cells) is, however, generated to extend the array only if an explicit write operation occurs - the extent of the generation of new cells then depends on the nature of the elements on the dimension in question (i.e. these may be atomic cells or arrays on a bounded or unbounded dimension) and will be the minimum permitted. Further generation will occur to fill any "holes" in the structure. See above for the initial content of all newly generated cells.

The minimum existence of an array on an unbounded dimension is one (minimum) element at the origin (coordinate value = 1).

The elements on one dimension need not be of the same lower-order dimensional structure where the lower-order dimension is unbounded; for example, if the dimensions are named Z, Y, and X with X the lowest-order dimension, and coordinate values are indicated as Y1, Y2, etc., and Y is unbounded, then the element at Z5 could consist of elements having Y coordinates Y1, Y2, and Y3, and the element at Z6 could consist of elements having Y coordinates Y1, Y2, ..., Y5.

2.2.3.1.2 Number of Dimensions

Although the concept of dimension hierarchy is open ended the number of dimensions is limited in Basic Class to 3 and these are known as Z, Y and X, with X the lowest-order. The parameterization capabilities of Basic Class allow a particular virtual terminal/ presentation connection to use 1 (X only), 2 (X and Y only) or 3 (X and Y and Z) dimensions and for any combination of bounded and unbounded to be defined.

2.2.3.1.3 Addressing the Dimension Structure

A set of coordinate values, one for each dimension in use, is known as a pointer and uniquely designates a single cell in the dimension structure. Basic Class provides a single pointer known as The Pointer.

Various methods are described below for setting and adjusting the value of this pointer, thus allowing an individual cell to be addressed to give access to the atomic object which may be associated with (contained within) it. This is known as position control.

2.2.3.1.4 The Move Window

The move window of a dimension identifies at any instant the lowest value of the range of coordinate values at which cell contents may be changed.

In the general case of a 2 or 3-dimensional CDS, the move windows and the historical entry of cell data define a set of coordinate combinations to which cell data can be written.

The sizes of the move windows are agreed parameters in any existing dimensions. The default value is "any".

The existence of move windows and the consequential set of valid coordinate combinations at which cell content can be accessed does not imply that all or any of the position controls enable the full set to be used.

The general rules for the move windows are as follows:

- the upper boundary of the move window in a particular dimension is the maximum coordinate of the dimension at which an object has been accessed,
- the lower boundary of the move window is either the origin (coordinate value 1) or the upper boundary less the size of the move window for the dimension,

- as atomic objects are accessed beyond the current upper boundary a new value of the upper boundary is defined; also a new value for the lower boundary may be defined as the move window is brought forward,
- the move window on any dimension cannot move backwards (i.e. to lower values on the dimension) for a given value of higher-order dimensions.

Particular rules for the dimensions X and Y are given below:

- X-dimension: For different Y-values, the X move windows may have different upper and lower boundaries depending on the agreed X-move-window-size and the previous manipulation of cell contents that has taken place.
- X-dimension: For different Z-values, the Y move windows may have different upper and lower boundaries depending on the agreed Y-move-window-size and the previous manipulation of cell contents that has taken place.

The move window applies an overriding constraint on movements within the CDS for the purpose of update access, in addition to any due to constraints on the position controls themselves, see 2.2.3.2, in that although a position control may result in a coordinate value lower than the lower boundary of the current move window on a dimension, any attempt to access the cell at that point is Invalid.

A move window width of zero is valid and implies that "backwards" access is not possible. A width of "any" implies that all values from the current maximum back to the origin (coordinate value 1) are accessible.

2.2.3.2 Methods of Position Control

The following methods of position control, with the exception of sequential implicit, can be applicable to any or all of the available dimensions in any combination. Although in most respects the position control facilities of the dimensions are independent, there are certain special combinatory position controls which are applicable independently of other constraints. These are given under the applicable dimension in sub-clauses 2.2.3.3 etc., see also sub-clause 2.3.7.4.

2.2.3.2.1 Sequential Implicit

This method is available always and only on the lowest-order dimension X. It is related to the accessing of the atomic objects on the X-dimension so that whenever such an object is accessed (to update some or all parts of it) the X-coordinate is then incremented

by 1 so that the next access will be to the next atomic object of the ordered array, if any.

If the X-coordinate was Xb then it becomes XL and in this case there cannot be a further atomic object and any attempt to access one is Invalid. XL can only be reached by this implicit position control.

In the case where X is Unbounded the X-coordinate can be incremented indefinitely. Explicit placing of a primary value in a position previously non-existing will generate cells at and up to the new position; this will move the window of X. See above for initial content of intermediate new cells.

2.2.3.2.2 Relative

This method is applicable to all dimensions but its availability and capability in a particular case is subject to negotiation.

It allows the dimension coordinate to be increased or decreased from its current value by a specified increment, and is subject to the following rules:

- a backwards move (negative increment) is not always available and, if not negotiated, is Invalid,
- positive and negative increments are subject to a maximum value, possibly different, declared at negotiation time, and larger values are Invalid,
- a negative increment which would result in a zero or negative coordinate, is Invalid,
- for a bounded dimension a positive increment which would result in a coordinate larger than the Bound is Invalid.

For an unbounded dimension the coordinate can be increased indefinitely but the move window is moved only if an object is generated beyond the maximum value at which one previously existed, i.e. beyond the current move window; the move window is then moved to align the upper boundary with the address of the new object.

2.2.3.2.3 Absolute

This method is applicable to all dimensions but its availability and capability in a particular case is subject to negotiation.

It allows the dimension coordinate to be set to a specified value, and is subject to the following rules:

- the new value is always greater than zero,

- a backwards move (new value less than the current) is not always available (declared capability at negotiation), and a smaller value is then Invalid,
- for a bounded dimension a new value greater than the bound is Invalid (limit cannot be set in this way).

For an unbounded dimension there is no limit to the coordinate value but the move window is moved only if an object is generated beyond the maximum value at which one previously existed, i.e. beyond the current move window; the move window is then moved to align the upper boundary with the address of the new object.

2.2.3.2.4 Home

This method is applicable to all dimensions but its availability and capability in a particular case is subject to the rule below.

Home enables the dimension coordinate to be set to the value 1, i.e. the origin.

Any use of Home which would result in a move which is invalid due to any other addressing rule is Invalid.

2.2.3.3 X-Dimension

This dimension is mandatory, and enables addressing of a one-dimensional ordered array of cells and atomic objects. Such a structure is formally known as an X-array.

X is considered to be horizontal where this is meaningful, and an X-array is commonly known as a Line, composed of Characters.

The X-dimension and X-array can be unbounded or bounded. If bounded the size of an X-array is fixed. If unbounded the size is variable (minimum 1) with new cells generated when required to store objects. Its current size is determined by the maximum value of the X-coordinate at which an atomic object exists, i.e. has non-nil Primary Value, or has ever existed. Intervening cells are initially "empty", i.e. primary value is "nil" and (applicable) character attributes have default values according to the PE. If Y is defined each X-array can have a different current size. An X-array cannot be reduced in size.

The following methods of position control are applicable and are subject to negotiation except as stated below to be "always available":

- Sequential implicit (always available)
- Relative
- Absolute
- Home.

Movement on the X-dimension never causes implicit move on any other dimension.

2.2.3.4 Y-Dimension

This dimension is not mandatory. When defined it represents in conjunction with X a two-dimensional array of (potential) storage cells, i.e. each element on the Y-dimension is an X-array. Such a structure is formally known as an X-Y-array. Y is considered to be vertical where this is meaningful, and an X-Y-array is commonly known as a Page, composed of Lines of Characters. If Y is not defined only one unique X-array can ever exist in the CDS.

An X-Y-array can be unbounded or bounded in the Y-dimension. If bounded the depth of a Y-array, in terms of number of X-arrays, is fixed. If unbounded, the depth of a Y-array is variable (minimum 1), with new X-arrays generated as required to store objects. Its current depth is determined by the maximum value of Y at which a non-nil X-array, (one with at least one non-nil object) exists or has ever existed. Intervening X-arrays, if X is unbounded, will initially have one empty cell. If Z is defined each Y-array may have a different current depth. Once generated the X-arrays continue to exist even though they may not be addressable within the position control capabilities of the VT. A Y-array cannot be reduced in depth.

The following methods of movement on the dimension are defined and are subject to negotiation:

- Relative
- Absolute
- Home.

A move to the start of the next X-array (y:=y+1, x:=1) is always permitted when the Y-dimension is defined regardless of whether any other relative or absolute addressing is or is not negotiated. Where Y is unbounded this will generate, if necessary, a new X-array (of minimum size).

Movement in the Y-dimension never causes implicit move on any other dimension.

2.2.3.5 Z-Dimension

This dimension is not mandatory; it can be defined when both X and Y are defined and in conjunction with X and Y represents an ordered set of X-Y-arrays. Such a structure is formally known as an X-Y-Z-array. If Z is not defined only one unique X-Y-array can ever exist in the CDS.

An X-Y-Z-array may where meaningful be considered as a group or book of pages. The X-Y-Z-array can be unbounded or bounded in the Z-dimension. If bounded, the number of X-Y-arrays (or thickness) of an X-Y-Z-array is fixed. If unbounded, the number of X-Y-arrays is variable (minimum 1), with new X-Y-arrays generated as required to store objects. Its current size is determined by the maximum value of Z at which an X-Y-array with at least one non-nil X-array exists or has ever existed. Intervening X-Y-arrays, if Y is unbounded will initially have a single X-array conforming to the rule for generating X-arrays in 2.2.3.2. Once generated such X-Y-arrays continue to exist even though they may not be addressable within the position control capabilities of the VT.

The following methods of movement on the dimension are defined and are subject to negotiation:

- Relative
- Absolute
- Home.

A move to the start of the next X-Y-array (z:=z+1,y:=1, x:=1) is always permitted when the Z-dimension is defined regardless of whether any other relative or absolute addressing is or is not negotiated. When Z is unbounded this will generate, if necessary, a new X-Y-array (of minimum size).

Movement in the Z-dimension never causes implicit move on any other dimension.

2.2.4 Global Attributes for the Basic Class

These attributes cover the global characteristics of the CCA of the Basic Class. The numbers in brackets after the headings are the reference numbers for the parameters shown in the directed graph in sub-clause 2.3.6.3.

2.2.4.1 Delivery Control (11.1)

The particularization of the delivery control facility described in GVT to Basic Class is that it is associated with the only CCA access control token available i.e. the write-access-token, and applies to the whole of the CCA, except for CSS areas that are defined as not subject to access control.

Permissible values: "no", "yes".

In this version the value of this parameter is implied by the operating subset (parameter 11, see also 2.3.2.1):

"VTB-A" implies "no",
all others imply "yes".

2.2.4.2 Erasure Capability (12)

There is no particularization to Basic Class other than implied by the nature of the atomic objects and their associated attributes. Erasure operates on Primary Values and may affect character attributes, see 2.3.7.5.

Permissible values: "yes", "no".

2.2.4.3 Control Information Encoding (13.1)

This parameter specifies how control information is encoded in the protocol. In Basic Class control information is encoded as embedded character strings according to ECMA-48.

Permissible values: "standard (ECMA-48) embedded", user-defined.

2.2.4.4 Graphic Information Encoding (13.2, 13.2.1, 13.2.1.1)

These parameters specify how graphic information is encoded in the protocol.

In Basic Class the graphic information is always represented as strings of octets, each octet containing either a 7-bit or an 8-bit code element, determined by p13.2:

Permissible values: "ISO-7", "ISO-8", user-defined.

The available graphic code elements are grouped into graphic coded sets of 94 code elements. Each graphic coded set is identified within the International Register of Coded Character Sets to be used with Escape Sequence, or by a private "name".

Parameters 13.2.1 and 13.2.1.1 specify the number of available coded sets and their names.

The available coded sets indicate the available character representation methods, e.g., if the supplementary coded set is available, a graphic can be represented by using a squence of non-spacing characters and a spacing character according to the so-called "composition" method, as defined in ISO 6937.

Note 1

The character repertoire specified by parameter 31.1.1 must be achievable by using the set or a subset of the elements of the available graphic coded sets.

2.2.5 Character Attributes (31)

This is the relevant atomic-object-type-attribute, see ECMA-87, for Basic Class. The CDS can have associated with it one or more Character Attributes. This implies that each Cell has the capability to store, at least conceptually, an Attribute Value for each of these attributes. The range of values available in the presentation environment in use, from the permitted set of values given in this Standard, is negotiated for each parameter and the result of the negotiation recorded in the DSD as part of the definition of the PE. The range of available values is the same for the whole of the CDS although the current value for individual cells can vary across the CDS.

The following list covers the character attributes defined in this version of Basic Class; this list may be added to in later versions as new device capabilities arise.

In the list below, the agreement of relevant capability is in terms of the repertoire of different values stored in the CDS which represent logical capabilities. The actual realization of these attributes is negotiated with, and is specific to, each Virtual Device.

In each case variability is available on a character-by-character basis.

2.2.5.1 Character Repertoire and Font Capabilities (31.1 etc.)

Parameter 31.1, Character Repertoire Capability, specifies how many repertoires are used; permissible values are 1, 2, ..., n.

Parameter 31.1.1 occurs once for each such repertoire if it is desired to associate semantic significance with them at the transfer syntax level. Entries in the sequence of multiple values may be "void" if merely acting as place holders for associating fonts with repertoires. "Void" values and all values if the parameter is omitted altogether are "unnamed" repertoires so far as the transfer syntax is concerned and the semantic significance is only associated with them at the realization level.

For each repertoire one or more fonts may be used (one is default) as specified in the associated 31.1.1.1 parameter; permissible values are 1, 2, ..., n. These too can be named at this level or left open until the realization level. Parameter 31.1.1.1.1 caters for naming them.

Permissible values for parameters 31.1.1 and 31.1.1.1.1 are names of standard repertoires/fonts, user names for user defined repertoires/fonts, or "void" as place holders for "unnamed" repertoires/fonts.

2.2.5.2 Emphasis Capability (31.2, 31.2.1)

Parameter 31.2 (permissible values 1, 2, ..., n) specifies how many levels of emphasis are required.

Parameter 31.2.1 (permissible values: the names of the emphasis attributes from ECMA-48 plus user-defined names) optionally allows the semantic significance of these to be specified for the transfer syntax. Otherwise this is left open until realization.

2.2.5.3 Colour Capability (31.3, 31.3.1)

Parameter 31.3 (permissible values 1, 2, ..., n) specifies how many colours are required.

Parameter 31.3.1 (permissible values: the colour names from ECMA-48 plus user-defined names) optionally allows the semantic significance of these to be specified for the transfer syntax.

2.2.5.4 Background Colour Capability (31.4, 31.4.1)

In a precisely similar way as parameters 31.3, 31.3.1 cater for foreground colours, these parameters cater for background colours.

2.2.6 CSS Area Descriptors

CSS Areas as described in GVT are available in Basic Class and may be utilised to hold Control, Signalling or Status information. Any such areas required over and above the primitive ones automatically created for each virtual device must be negotiated prior to their attachment to specific virtual devices (see below).

2.2.7 Virtual Devices

It is possible to use VTS Basic Class for restricted forms of application to application communication in which case no virtual devices need be defined. However normally, following agreement on the CCA, one or more virtual devices will be negotiated for association with the CCA. The negotiation for each virtual device requires, at a minimum a virtual device identity to be agreed. In addition the series of parameters described below allows the realization

of logical capabilites previously defined for the CCA to be agreed or controlled.

2.2.7.1 Virtual Device Identifier (71)

This is a mandatory parameter and has no default.

Permissible values: any user-defined character string unique to the PE.

2.2.7.2 Initial Status ON/OFF (71.1)

This parameter defines the initial status of the device.

Permissible values: "on", "off".

2.2.7.3 Access Control on Default CSS Area (71.2)

This parameter allows the access control over a Default CSS Area to be agreed.

Permissible values: "controlled", "uncontrolled".

2.2.7.4 Repertoire and Font Realization (74.1, 74.1.1)

These parameters allow the realization of repertoire and font to be controlled. They will override, in the case of conflicting corresponding values, any semantic significance established for the transfer syntax by parameters 31.1.1 and 31.1.1.1.1. Permissible values are registered repertoire/font names, user-supplied names and "void". No repertoire may be left "void" in both 31.1.1 and 74.1. If font is left "void" in both 31.1.1.1.1 and 74.1.1 then "as available" is assumed.

2.2.7.5 Emphasis Realization (75)

This parameter allows the realization of emphasis levels to be specified. It will override a specification for transfer syntax semantic in 31.2. If "void" or absent 31.2 applies. The default is "as available".

Renditions can be grouped into the following categories:

- Intensity (normal, high, low)
- Blink (steady, fast, slow)
- Image (positive image, negative image)
- Secret (revealed, concealed)
- Underline (no, single)

It is permissible to combine at most one available rendition from each category. Renditions within a category cannot be combined. For further information on a description of these terms refer to ECMA-48.

2.2.7.6 Foreground Colour Realization (76)

This parameter allows the realization of colour to be specified. It will override any specification for transfer syntax semantics in 31.3.1. If "void" or absent 31.3.1 applies if specified. If both are "void" the standard sequence of ECMA-48 is default (as many as specified from) "black", "red", "green", "yellow", "blue", "magenta", "cyan", "white". These are also the permissible values for the parameter.

2.2.7.7 Background Colour Realization (77)

In a precisely analogous way to 76 this parameter controls colour realization for background.

2.2.7.8 Layout Control (78, 78.1, 78.2)

These parameters allow the minimum realizable numbers of contiguous cells on the X and Y dimensions of the virtual device to be agreed.

Permissible values (in each dimension): n, "unbounded" (see below for defaults).

Where so agreed for a particular dimension, contiguous values within the dimension up to the agreed value will be presented contiguously on the virtual device. Beyond the agreed values no such commitment pertains. If the dimension is bounded the default is the bound. If the dimension is unbounded the default is unbounded. Thus the default always offers distortion-free layout.

Note 2

In this version, Basic Class provides only the above explicit control of layout. It does not provide explicit control of the size of rendition of atomic objects, or of, for example, inter-object gap, overlap or spacing, or of whether cell backgrounds are continuous, spaced or overlapped, and whether obscuration or combination occurs in the event of overlap of foregrounds or backgrounds. Such facilities may be included in a later version and/or in more advanced classes.

2.2.7.9 CSS Area Realization (79, 79.1, 79.2)

These parameters enable CSS areas to be attached to a virtual device and to be given a specific realization.

Permissible values for each named CSS data area: "signal", "status", "device control".

Further interpretation of these data areas is outside the scope of this Standard.

The default CSS area is an exception to this and is defined in sub-clause 2.3.9.1.

2.3 Service Description

2.3.1 Introduction

This Clause describes the services offered to the presentation-service-users in the particular case of Basic Class VTS operating as a specific presentation service of GPS. It does this in terms of elementary operations, known as Service Elements, on a common conceptual service interface. The concept of such an interface is necessary for this description but no conformance rules are applied to it. A service Element consists, in general, of a sequence of service primitives on the conceptual service interface, a service primitive, usually abbreviated to just primitive, being an indivisible event on one service access point.

For further information and guidance on the relationship between this service interface and the protocol which is defined in Section 3, refer to ECMA-87. For further information on the generic services in presentation layer refer to ECMA-86.

This service description is in terms of the Model of the Virtual Terminal described in Clause 2.2 of this Standard. It refers extensively to the GVT, ECMA-87, which itself refers to the GPS, ECMA-86.

The following material first identifies the major aspects of the VTS which can be covered by reference to the above mentioned Standards ECMA-86 and ECMA-87, and then in the later subclauses gives more detail on those aspects of the service which are specific to Basic Class.

Terminology and Description Technique for Service Elements is described in Appendix C of this Standard.

2.3.2 Functional Phases, Facilities and Subsets of Basic Class VTS

The functional phases of the VTS for Basic Class conform to the GVT; this refers to the description of the functional phases given in sub-clause 2.2.1 of GPS, and reference should be made to ECMA-86. Because Basic Class VTS allows only a single Defined Presentation Environment no use is made of the Multiple-PE option. Both levels of negotiation facility are available for selection at connection establishment time.

2.3.2.1 Service Subsets

A hierarchical set of service subsets are defined for VTS Basic Class. These progressively add service facilities as given below.

In this version of this Standard all the subsets make provision for a single Presentation Environment only.

If the operating subset agreed at connection establishment time is "VTB-A", "VTB-B" or "VTB-C" the PE must be fully defined at this time so that the transfer-enclosure, which is the only enclosure provided with these subsets, can be entered directly from the establishment phase. With operating subset "VTB-C" changes can be made subsequently by single-interaction negotiation. With subset "VTB-D" the above constraint does not apply, null- and negotiation-enclosures being available.

The operating subset is not inherently fixed for the p-connection, but "VTB-A" and "VTB-B" in fact provide no means of changing the operating subset once agreed at establishment time.

In this version of this Standard the pe-save/resume capability included in GPP, ECMA-86, is not available.

VTB-A: Kernel Subset

provides: P-CONNECT
P-RELEASE
P-DISCONNECT
P-ABORT
P(VT)-GIVE-TOKENS
P(VT)-REQUEST-TOKENS
P(VT)-DATA
P(VT)-CSS
P(VT)-DELIVER
P(VT)-ACK-RECEIPT } no delivery-control
P(VT)-FREECSS

This subset uses the facilities of GPP subset "GP-A". The following optional facilities of GP-A are supported in VTB-A:

token for mediating use of Release : yes

token for mediating Normal Data flow in transfer-enclosure : yes

token for mediating use of Change Enclosure : not applicable

token for mediating use of Perform Negotiation : not applicable.

VTB-B: Basic Extension Subset

Provides no services in addition to services of VTB-A. This subset adds the delivery-control feature to VTB-A.

This subset uses the facilities of GPP subset "GP-A". The following optional facilities of GP-A are supported in VTB-B:

token for mediating use of Release : yes

token for mediating Normal Data flow in transfer-enclosure : yes

token for mediating use of Change Enclosure : not applicable

token for mediating use of Perform Negotiation : not applicable.

VTB-C: Simple Negotiation Subset

Provides, in addition to facilities of VTB-B:

P-PERFORM-NEGOTIATION

This subset uses the facilities of GPP subset "GP-B". It enables the PE to be changed from within the transfer-enclosure using the above service element. The following optional facilities of GP-B are supported in VTB-C:

token for mediating use of Release : yes

token for mediating Normal Data flow in transfer-enclosure : yes

token for mediating use of Perform Negotiation : yes

token for mediating use of Change Enclosure : not applicable

VTB-D: Full Negotiation Subset

Provides, in addition to facilities of VTB-C:

P-CHANGE-ENCLOSURE
P-INVITE
P-OFFER
P-ACCEPT
P-REJECT

This subset uses the facilities of GPP subset "GP-C". It enables establishment of a p-connection with an incomplete PE, with the use of any of the negotiation facilities to complete this before use in transfer-enclosure or to amend it at any subsequent time. The following optional facilities of GP-C are supported in VTB-D:

P-CLOSE/ABORT-ENCLOSURE : no

token for mediating use of Release : yes

token for mediating Normal Data flow in transfer-enclosure : yes

token for mediating use of Perform Negotiation : yes

token for mediating use of Change Enclosure : yes.

Further subsets to add Multiple-PE operation and Restart facilities are for further study for inclusion in a later version.

2.3.3 Establishment and Termination of Presentation-Connection

2.3.3.1 Establishment of Presentation Connection (p-connection)

The general description of this phase given in GVT is applicable to Basic Class. Details of parameters, etc. specific to Basic Class are given below.

The service elements applicable are as follows; also given is the Basic Class specific parameter usage. Aspects not mentioned conform to the standard descriptions in GVT or, where referenced by it, GPS.

P-CONNECT

Parameters with Basic Class specific usage aspects:

p-service-class-ident: applicable value is "basic".

p-service/class-version-idents: applicable value for this version of Basic Class is 1. Other values Reserved.

p-service/class-subset-ident: applicable values are defined in sub-clause 2.3.2.1.

p-pe-profile: as GVT; applicable profile names and definitions standardized in this version of Basic Class are given in sub-clause 2.3.6.

p-special-conventions: available as GPS, and GVT, but only as a component of p-fixed-pe-params; usage and meaning and any effect on other parameters of this Standard is user-defined.

p-save-pe-capab: applicable value is "no".

p-max-pes: automatically defaulted to 1 in this class as multiple-pe capability is not provided.

Additional information

The usage of GPS/P subset is implied by the class subset as defined in sub-clause 2.3.2.1. There is no inherent restriction on re-negotiation of the Basic Class subset and hence of this usage, but refer to 2.3.2.1. The rules given in GVT, apply to other parameters of P-CONNECT.

The session service parameters implied from the P-CONNECT service parameters are given in Section III.

The maximum length of the content of p-transp-data parameter is 36 octets; use of p-special-conventions will reduce this by (2+length of the content of that parameter).

2.3.3.2 Termination of Presentation Connection

The following service elements are applicable to this facility and are as described in GVT, including parameter usage, except where qualified by Basic Class specific information below.

P-RELEASE

P-DISCONNECT

P-ABORT

Class specific usage aspects

In Basic Class ownership of the p-terminate-token which conditions the use of P-RELEASE is always the same as for all the other service-visible tokens; no separate explicit control of the p-terminate-token is provided in Basic Class. P-RELEASE is negotiable, i.e. the request can be refused.

2.3.4 Token Control Facilities

The following service elements are applicable to this facility and are as described in the GVT, including parameter usage, except where qualified below by general Basic Class specific information, and by enclosure-specific information in later sub-clauses.

Within the Basic Class the tokens that control the service are at any time all owned by one and the same p-user; i.e. separate control is not possible.

P(VT)-GIVE-TOKENS

Class specific parameter usage

p-token-ident parameter is not relevant; in Basic Class all the service-visible tokens are always owned together no separate control being provided over any one token. p-encl-service-specific parameter is not used.

P(VT)-REQUEST-TOKENS

Class specific parameter usage

p-token-ident parameter is not relevant; in Basic Class all the service-visible tokens are always owned together, no separate control being provided over any one token.

2.3.5 Enclosure Control Facility

The following service element is applicable to this facility and is as described in GVT. Basic-Class-specific information on parameter usage is given in later sub-clauses on negotiation and transfer enclosure operation.

P-CHANGE-ENCLOSURE

P-CLOSE/ABORT-ENCLOSURE is not available in this version.

2.3.6 Negotiation Facilities, Parameters and Profiles

Both levels of negotiation facility described in GVT by reference to GPS are available in Basic Class; the usage is implicit in the class-subset selected; there is no inherent restriction on changes to this by re-negotiation (if available in the PE as defined at any time). Basic Class makes provision for a single PE only.

The following service elements are applicable to this facility and are as described in the GVT, including parameter usage, except where qualified by Basic-Class-specific information, including class-subset in use.

2.3.6.1 Single-interaction Negotiation

P-PERFORM-NEGOTIATION

Class specific parameter usage

p-pe-ident, p-draft-pe: not relevant since there is only a single PE in Basic Class and no identifier is defined for it; the only available negotiation action is "replace" commencing with the "initial" PE.

p-pe-specific-params: can include p-service/class-subset-ident and p-pe-profile, see parameter descriptions under P-CONNECT in this Standard. Can also contain either or both p-param-ident-list, p-param-list, see 2.3.6.2; use of p-param-ident-list implies that any offer in the response/confirmation valid in the selected subset will be accepted (further negotiation can of course be initiated).

p-special-info: available as GPS; usage and meaning and any effect on other parameters of this Standard is user-defined.

2.3.6.2 Multiple-interaction Negotiation

The service elements provided by GVT for this level of negotiation facility are applicable as given below, subject to the subset in use. Class-specific information is given below:

P-CHANGE-ENCLOSURE

Class specific entry parameter usage

(p-dest-encl-type has value "negotiation" with this usage; refer to sub-clause 2.3.7.1 for use for entry to transfer-enclosure):

p-pe-ident, p-draft-pe: see 2.3.6.1.

p-initial-pe-params : can include p-service/class-subset and p-pe-profile, see parameter descriptions under P-CONNECT in this Standard.

p-special-info: available as GPS, usage and meaning and any effect on other parameters of this Standard is user-defined.

Class specific exit parameter usage

as GVT/GPS except that p-pe-action cannot have value "delete" (it is not permissible to discard the (only) PE in Basic Class).

P-NEG-INVITE
P-NEG-OFFER
P-NEG-ACCEPT
P-NEG-REJECT

Class specific parameter usage: p-param-ident-list, p-param-list; the detailed parameters for Basic Class are described in sub-clause 2.3.6.3.

P(VT)-GIVE-TOKENS

Class/enclosure specific usage aspects

Within a negotiation-enclosure the negotiate-token is the service token used to control the negotiation dialogue. Passing ownership of this using P(VT)-GIVE-TOKENS will also pass ownership of the enclosure-token and the p-terminate-token.

P(VT)-REQUEST-TOKENS is not available in negotiation-enclosure (since all service tokens are always held together in Basic Class and requesting the negotiate-token is not required).

2.3.6.3 Presentation Parameters for Basic Class - General

Sub-clause 2.3.6.4 lists the individual presentation-parameters used in Basic Class. Each parameter is given a reference number. References to a parameter will be by reference number (prefixed by the letter p if the context requires it). A compound parameter is used to associate a number of sub-parameters with one main-parameter. A sub-parameter is given a number of the form: m.n , where m is the main-parameter number and n is the sub-parameter number. There is no inherent limit to the depth of this nesting.

The following information is given for each parameter:

Reference Number

Parameter Name

Parameter Multiplicity

1 : parameter occurs once only with respect to each occurrence of its immediate parent,
M : parameter may occur more than once as above. Whether M = 0 is allowed is given in the "Remarks" column.

Parameter Type

S : Simple Value
C : Compound Value

Value Type

S : symbolic
M : bit map
N : numeric
C : character string
T : transparent

Notes on the meaning of these value types are given in section 3, (sub-clause 3.3.4.2), but the value type actually used in the protocol messages is not necessarily the same.

A directed graph showing the interdependency of the parameters is shown below. The following sub-clause gives all parameters in tabular form with appropriate details and notes.

Remarks on the Directed Graph

1. Only parameters which are actually negotiable in Basic Class are shown (below the service-class-version level). This implies that certain parameters shown in the skeleton graph in GVT which are "fixed" by virtue of class = "basic" have been omitted.

2. Any change to any parameter causes all descendents to revert to their appropriate default values.

3. Any change to a CSS descriptor is automatically reflected in any previously agreed instances of the CSS area attached to virtual devices.

4. The standard does not guarantee that the contents of the CDS are sensibly retained during re-negotiation of the following parameters: 13.2, 20.4, 21, 22, 23, 31.1, 31.2, 31.3, 31.4.

5. Minor differences between parameter names in the graph and the table in 2.3.6.4 are not significant.

6. M ⊨ Denotes that multiple occurrence is possible.

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

Directed Graph for Parameters in Basic Class

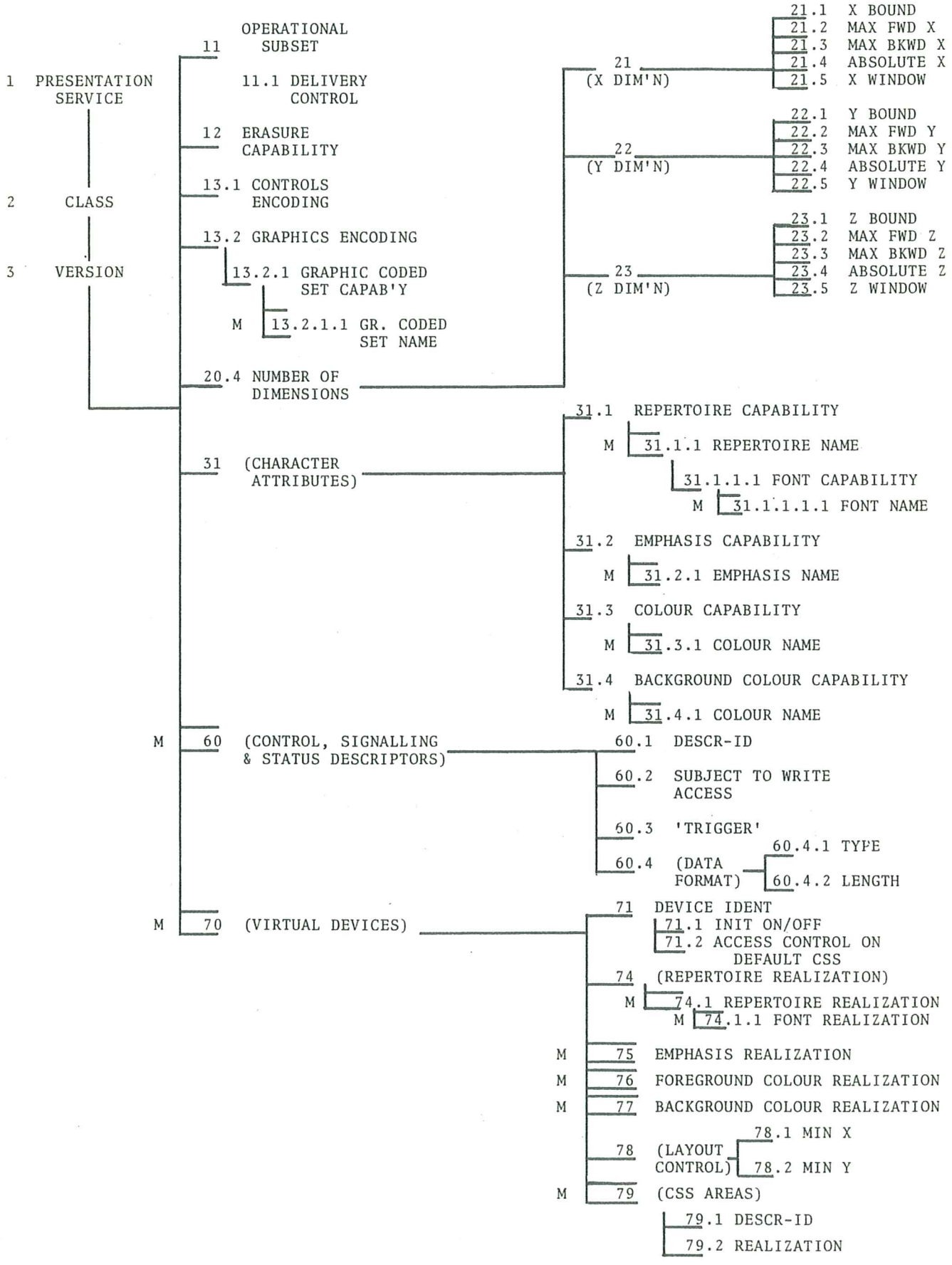


Table of Parameters for Basic Class

PARAMETER REFERENCE NUMBER	NAME	SINGLE OR MULTIPLE	PARAM TYPE SIMPLE OR COMPOUND	VALUE TYPE	SYMBOLIC VALUES	NOTES AND REMARKS
1	presentation-service	1	S	S	"VTS"	(1), (2)
2	VTS-class	1	S	S	"basic"	(1), (2)
3	version	1	S	S	"1"	version 1 (1), (2)
11	operating-subset	1	S	S	"VTB-A", "VTB-B" "VTB-C", "VTB-D"	
11.1	delivery-control	1	S	S	"no", "yes"	
12	erasure-capability	1	S	S	"no", "yes"	
13.1	control-encoding	1	S	S	"standard embedded" user-defined	ECMA-48 is the relevant standard in this version.
13.2	graphics-encoding	1	S	S	"ISO-7", "ISO-8", user-defined	ECMA-6, ECMA-35 and ECMA-43 are relevant in this version
13.2.1	graphic-coded-set-capab'y	1	S	N	1, 2, ..., n	
13.2.1.1	graphic-set name	M	S	C	according to Int. Register	
20.4	number-of-dimensions	1	S	N	1, 2, 3	
21	X-dimension	1	C	-		
21.1	bound	1	S	S	"unbounded", n	$n \geqslant 1$
21.2	max-forward-rel-move	1	S	S	0, 1, ..., n, "any"	(3). Forward movement is assumed normal.
21.3	max-backward-rel-move	1	S	S	0, 1, ..., n, "any"	(3). Backward movement is assumed exceptional.
21.4	absolute-move	1	S	S	"none", "higher", "any"	(3).
21.5	window-width	1	S	S	0, 1, ..., "any" (see remarks for default)	0 is default for unbounded case, "any" is for bounded case.
22	Y-dimension	1	C	-		
22.1	bound	1	S	S	"unbounded", n	See 21.1.
22.2	max-forward-rel-move	1	S	S	0, 1, ..., n, "any"	(3). See also 21.2.
22.3	max-backward-rel-move	1	S	S	0, 1, ..., n, "any"	(3). See also 21.3.
22.4	absolute-move	1	S	S	"none", "higher", "any"	(3).
22.5	window-width	1	S	S	0, 1, ..., "any" (see remarks for default)	0 is default for unbounded case, "any" is for bounded case.
23	Z-dimension	1	C	-		
23.1	bound	1	S	S	"unbounded", n	See 21.1

PARAMETER REFERENCE NUMBER	NAME	SINGLE OR MULTIPLE	PARAM TYPE SIMPLE OR COMPOUND	VALUE TYPE	SYMBOLIC VALUES	NOTES AND REMARKS
23.2	max-forward-rel-move	1	S	S	0, 1, ..., n, "any"	(3). See also 21.2
23.3	max-backward-rel-move	1	S	S	0, 1, ..., n, "any"	(3). See also 21.3
23.4	absolute-move	1	S	S	"none", "higher", "any"	(3).
23.5	window-width	1	S	S	0, 1, ..., "any" (see remarks for default)	0 is default for unbounded case, "any" is for bounded case.
31	character-attributes	1	C	-		
31.1	repertoire-capability	1	S	N	1, 2, ..., n	
31.1.1	repertoire-name	M	S	C	registered name, user defined name, "void"	(7), (8). Default is parameter omitted which implies n unnamed repertoires. "void" is used as a place holder for an unnamed repertoire when multiple fonts are required.
31.1.1.1	font-capability	1	S	N	1, 2, ..., n	may appear once for each repertoire, specifies how many fonts for each.
31.1.1.1.1	font-name	M	S	C	registered font name, user defined name, "void".	(7). Default is parameter omitted which implies n unnamed fonts for this repertoire.
31.2	emphasis-capability	1	S	N	1, 2, ..., n	number of logical (distinguishable) emphasis levels required.
31.2.1	emphasis-name	M	S	C	ECMA-48 emphasis realization names	(7). Default is parameter omitted, implies n unnamed emphasis levels.
31.3	colour-capability	1	S	N	1, 2, ..., n	Applies to foreground colour.
31.3.1	colour-name	M	S	C	ECMA-48 colour realization names	(7). Default is parameter omitted, implies n unnamed colours.
31.4	background-colour-capability	1	S	N	1, 2, ..., n	
31.4.1	background-colour-name	M	S	C	ECMA-48 colour realization names	(7). See also as for 31.3.1
60	CSS-descriptors	M	C	-		"none" is default, but see note (4).
60.1	descriptor-ident	1	S	C	user defined	Mandatory for each CSS descriptor (5).
60.2	subject-to-write-access	1	S	S	"yes", "no"	Token does not need identifying (Basic Class has only one).
60.3	forces-give-token ("trigger")	1	S	S	"no", "yes"	"yes" implies that writing to the CSS area implies P-GIVE-TOKEN.
60.4	data-format	1	C	-		
60.4.1	data-type	1	S	S	"C","N","S","T","M"	(6)
60.4.2	field-length	1	S	N	1, 2, ..., n	Specifies number of octets.

PARAMETER REFERENCE NUMBER	NAME	SINGLE OR MULTIPLE	PARAM TYPE SIMPLE OR COMPOUND	VALUE TYPE	SYMBOLIC VALUES	NOTES AND REMARKS
70	virtual-devices	M	C	-		
71	virtual-device-ident	1	S	C	user defined	Mandatory for each virtual device (5)
71.1	initial-on/off	1	S	S	"on", "off"	Indicates whether virtual device is initially "on" or "off".
71.2	access-contr-on- default-CSS	1	S	S	"yes", "no"	
74	repertoire-realization	1	C	-		
74.1	repertoire-realization	M	S	C	registered name, user defined name, "void"	(8). May be omitted or "void" only if corresponding entry in 31.1.1 is non-void in which case it defaults to the same as the corresponding 31.1.1 name.
74.1.1	font-realization	M	S	C	registered font name, user defined name, "void"	(9). See as for 74.1 with respect to entries in corresponding 31.1.1.1.
75	emphasis-realization	M	S	C	ECMA-48 emphasis realization names,...	(10). Default is Standard ECMA-48.
76	colour-realization	M	S	C	ECMA-48 colour names, ...	(10). Default is Standard ECMA-48.
77	background-colour-realization	M	S	C	ECMA-48 colour names, ...	(10). See as for 76.
78	layout-control	1	C	-		
78.1	min-X	1	S	S	n, "unbounded"; see remarks for default	If dimension is bounded, the default is the bound, if unbounded, default is unbounded.
78.2	min-Y	1	S	S	As 78.1	As 78.1.
79	CSS-area	M	C	-		
79.1	descriptor-ident	1	S	C	entries in 60.1 above	(5).
79.2	realization	1	S	S	"status", "device control", "signal"	(6).

Notes on Parameter Graph and Table
(quoted in parentheses in the table)

1. This parameter is "fixed" for the p-connection if it is specified (directly or indirectly through a profile) in p-pe-fixed-params at establishment time. In this version this must be the case for parameters p1, p2 and p3.

2. This parameter is not negotiable - its value is established (directly or indirectly through a profile) on entry to negotiation (both single- and multiple- interaction types). If its value has not been "fixed" at establishment (see 1 above) it is set by its earliest appearance in the sequence (profile in p-initial-pe-params), (p-initial-pe-params), (profile in p-pe-specific-params), (p-pe-specific-params).

3. Regardless of limits negotiated on addressing capability any actual movement which violates any bound will be diagnosed as illegal.

4. The agreement on a Virtual Device ident automatically brings into existence a default CSS area with the same name as the device with default for the other parameters and a data format of 1 octet bitmap of which the first 4 bits have the following significance:

   - ON/OFF control
   - interrupt
   - status alert
   - alarm.

5. These parameters are not negotiable, i.e. do not appear in lists of offers and cannot be counter-offered.

6. The user definition capability for CSS areas in this version is limited to uniform type in 60.4.1 and high level realization in 79.2. A later version may add a structure capability to p60.4 and will add appropriate realization specification parameters. The default CSS area automatically supplied for each device is already composite with the 4 defined bits corresponding to CONTROL (1 bit) and signals (3 bits).

7. These parameters are all optional. If specified they have the conceptual effect of tying semantic significance to the attribute values in the CDS. They may be over-ridden if desired by a conflicting assignment in the corresponding positions in the realization parameters. If they are omitted then the p-user only assigns logical distinguishability to the values

in the CDS leaving any semantic significance to be specified (if desired/essential) through the realization parameters.

8. Repertoire name must be specified for each repertoire in the set either in 31.1.1 or in 74.1. No standard significance is defined if corresponding entries conflict, however it should be possible to use, for example, IRV in 31.1.1 and IRV/UC in 74.1 and expect folding to occur satisfactorily.

9. If font name is not supplied in either position or is "void" then "as available" is assumed as default.

10. The .... indication in the symbolic values is intended to show that conceptually any attribute realization name may occur here (existing standard ECMA-48, or future extensions to that, or user-defined names) so that there is no conceptual reason why emphasis should not be realized by colour or colour by grey-scaling and the like. However this version of the Standard does not define or guarantee that anything outside the colour scope, font scope, emphasis scope, etc., of ECMA-48 will work in a predictable way or can be successfully negotiated.

2.3.6.5 Standard Profiles for Basic Class

This Clause gives the list of Standard Profiles for this version of Basic Class. The profiles given in this Standard, including the "local" names, are meaningful and valid only in the context of "ECMA.VTS.Basic"; thus a full identification of these profiles outside the context of this Standard document would need to include this outer context identification.

Note 3

The addition of some further profiles is likely in subsequent versions, and there is likely to be a need for a scheme for the registration of standard profiles with some appropriate registration body. When this question is resolved it is possible that profile definitions will all appear in some other place than this Standard and that this clause will then be replaced by a standard format for profile descriptions.

2.3.6.5.1 Introduction to Profile Concept

Virtual Terminal Profiles provide a mechanism by which the parameters of certain commonly encountered virtual terminals can be directly agreed without the need for full explicit parameter negotiation.

A profile defines a complete (i.e. usable) set of p-connection parameters (within the established context of "ECMA.VTS.Basic", see note in 2.3.6.5).

The p-user receiving the offer of a profile, which offer must include the selection of values for the mandatory parameters of the profile, see below, may in general, either accept, or reject, the offer. There is no method of individual parameter negotiation during the agreement of a profile. The single-interaction-negotiation, however, does allow individual parameters to be agreed at the same time; these may alter the profile but must remain consistent with it.

Once a PE has been agreed by means of a profile, re-negotiation of individual parameters may be performed in the normal manner (except where "fixed" at connection establishment time, and assuming that the current operating subset includes negotiation facilities).

In order to accommodate commonly encountered variations of device types, a number of mandatory parameters are associated with each profile type. Whenever a profile is offered values for all such parameters must be defined. These have, in general, equivalents in the list of negotiable parameters given in section 6.4.1, but are designated in the profiles by the letter "r" and the identifiers and values are not the same.

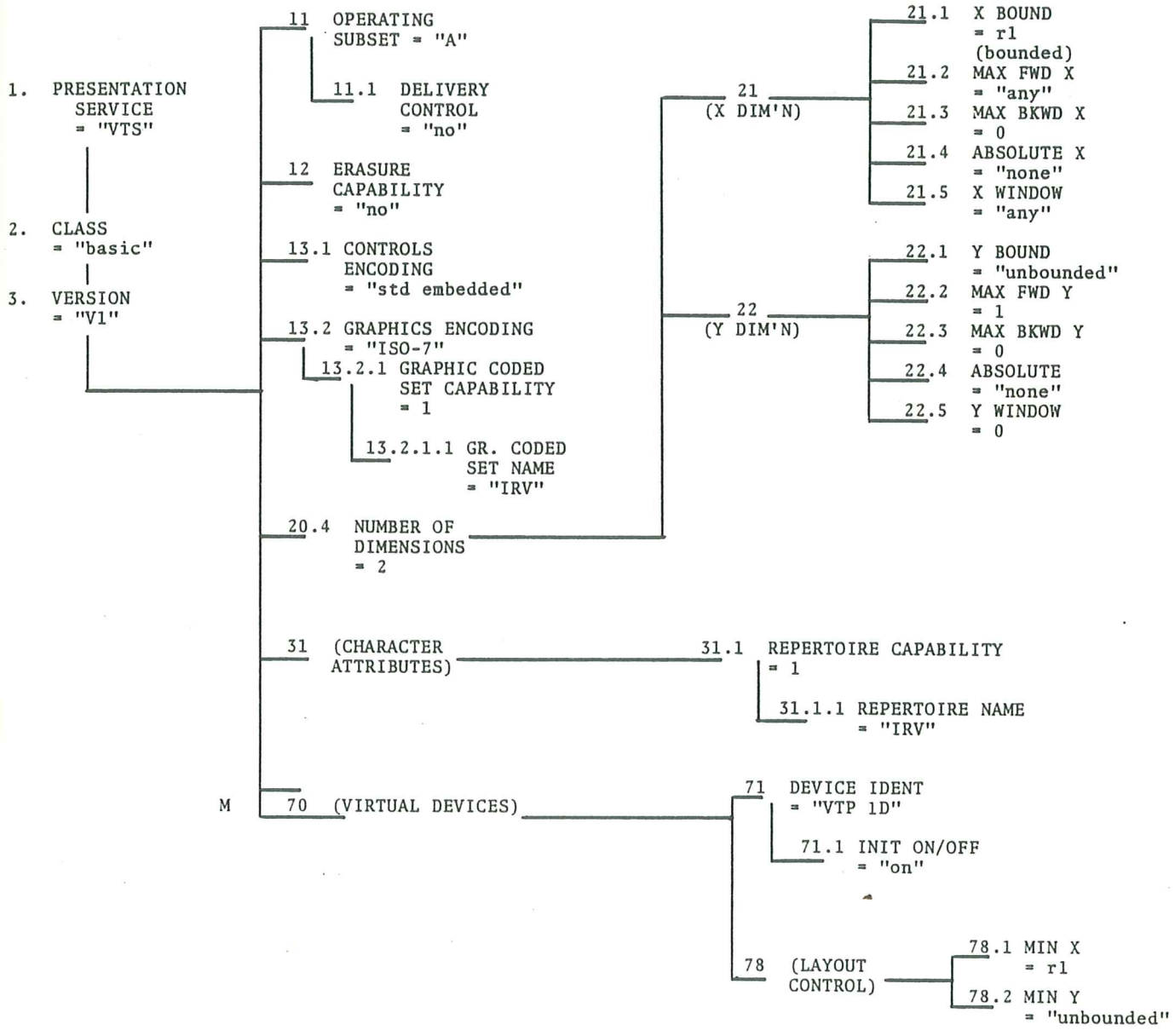
Profiles are identified symbolically by a number, with a capital P preceding it, and these may be taken as the valid values for use (only) in the service parameters of P-CONNECT, P-PERFORM-NEGOTIATION and P-CHANGE-ENCLOSURE. (Outside the context of this Standard some further, more global, identification scheme will be needed; see note in 2.3.6.5).

2.3.6.5.2 Standard Profiles for Basic Class

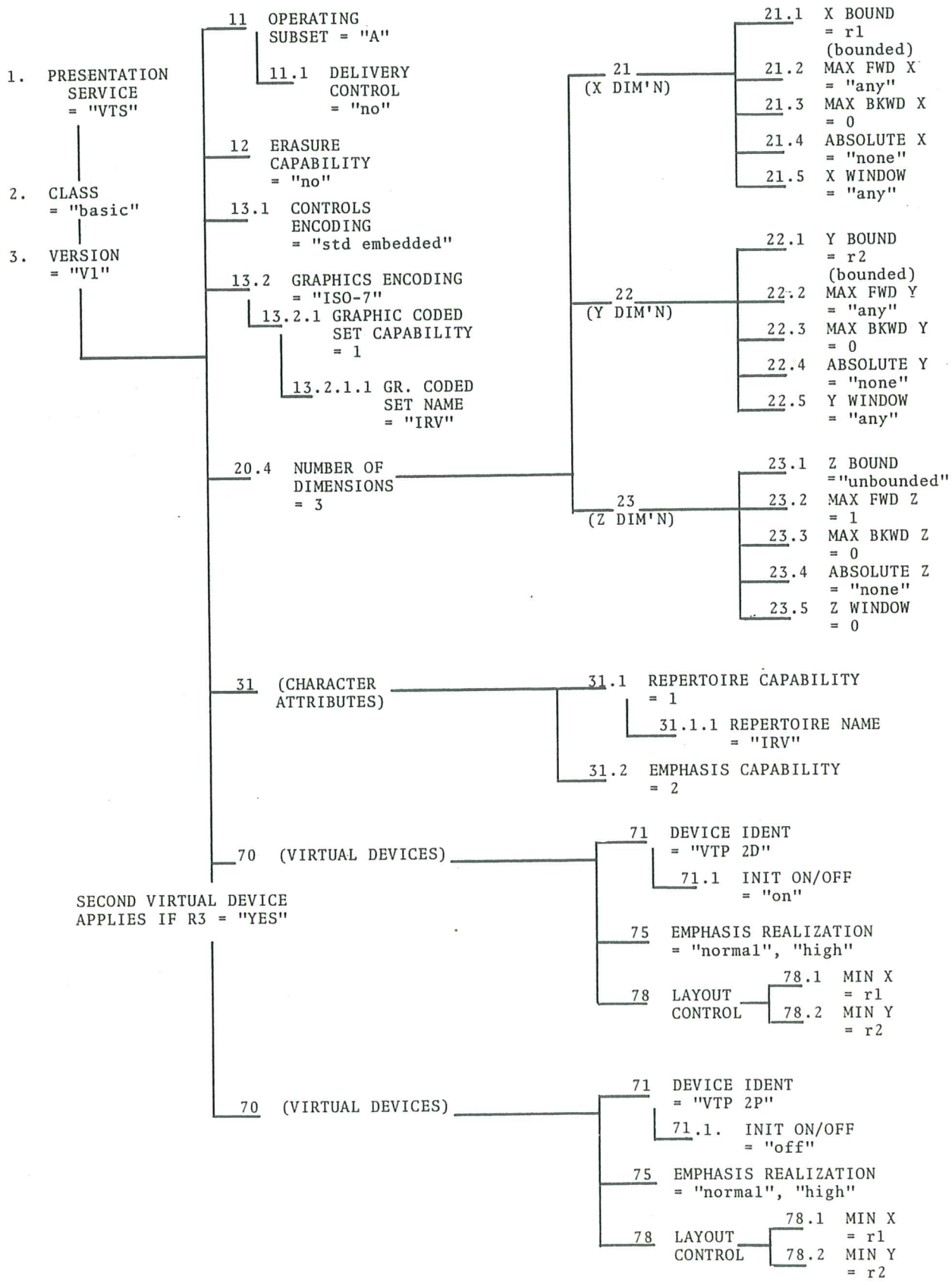
Four Standard Profiles are defined for this version of Basic Class and correspond to "logical" line-mode-relative-addressing terminals (P1, e.g. TTY like), page-mode-relative-addressing terminals (P2, simple VDUs, teleprinters, printers), and "full-screen"-page-mode terminals with fixed-repertoire monochrome capability (P3) or multi-repertoire and colour capability (P4).

The Standard Profiles are defined formally by particular instances of the Directed Graph of sub-clause 2.3.6.3 with appropriate values inserted and unused parameters omitted. These are given below for the profiles P1 through P4.

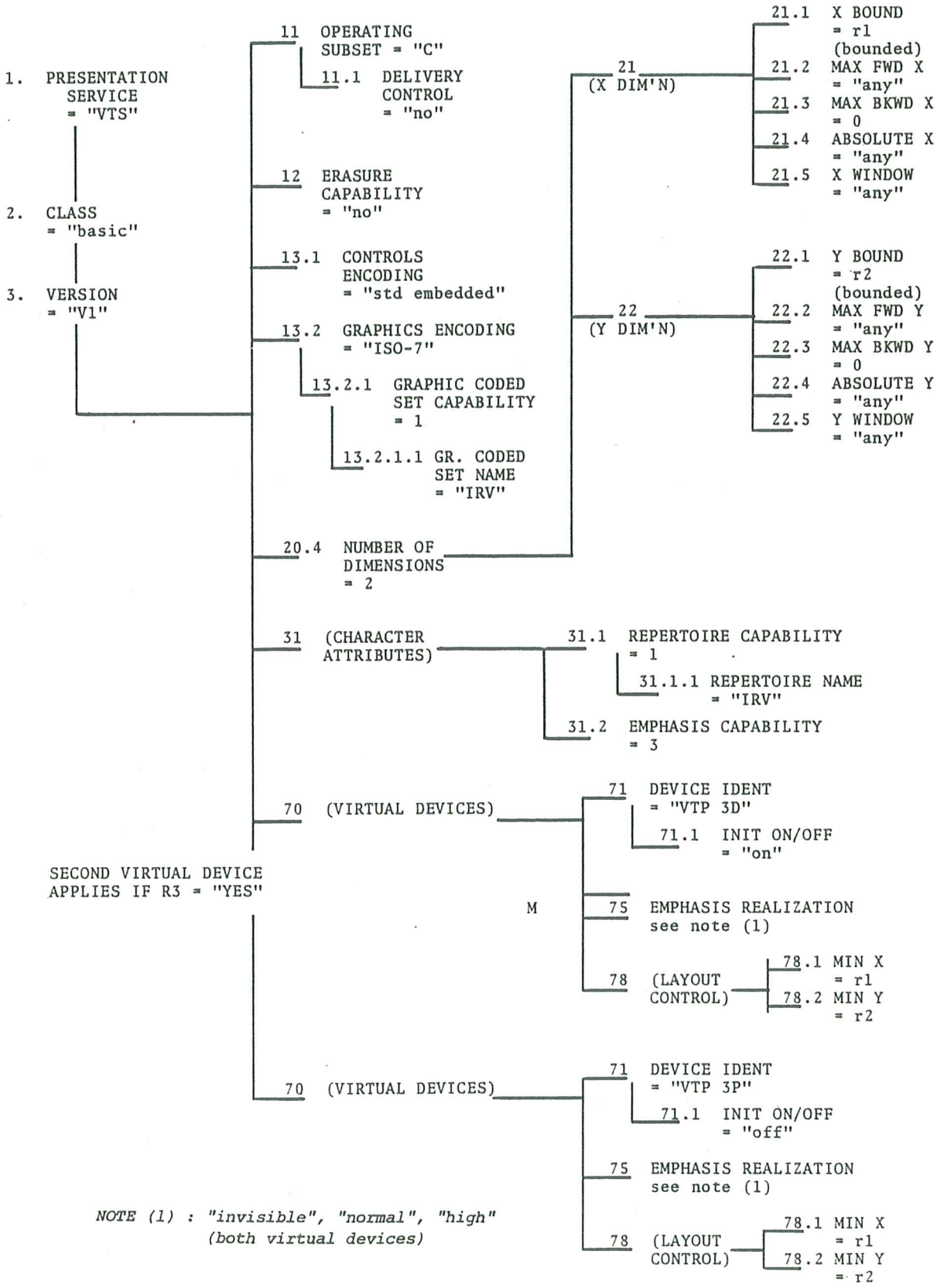
Standard Basic Class Profile P1 (r1)



Standard Basic Class Profile P2 (r1,r2,r3)

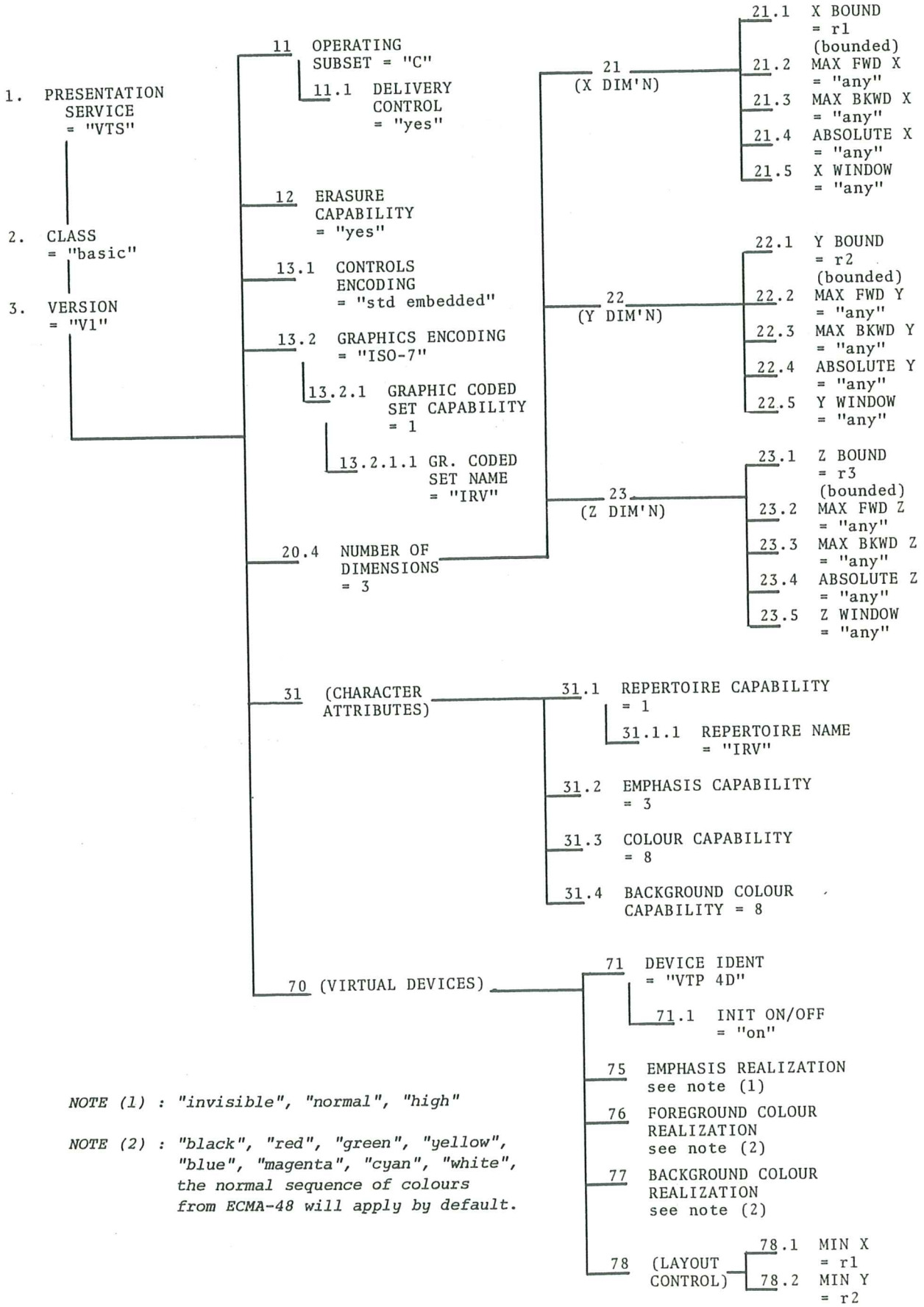


Standard Basic Class Profile (P3 (r1,r2,r3)



NOTE (1) : "invisible", "normal", "high"
(both virtual devices)

Standard Basic Class Profile P4 (r1,r2,r3)



NOTE (1) : "invisible", "normal", "high"

NOTE (2) : "black", "red", "green", "yellow", "blue", "magenta", "cyan", "white", the normal sequence of colours from ECMA-48 will apply by default.

2.3.7 Information Entry and Presentation to Other P-User

This phase conforms to the general statements for this phase given in the GVT. Basic Class does not use the Multiple-PE option, i.e. only one PE is used. The following material is Basic Class dependent detail.

2.3.7.1 The Transfer Enclosure

The generic description of entry to and exit from transfer-enclosure given in GVT is applicable with the class-dependent qualification below.

P-CHANGE-ENCLOSURE

Class specific entry parameter usage
(p-dest-encl-type takes value "transfer" with this usage; refer to 2.3.6.2 for usage for entry to negotiation-enclosure)

p-pe-ident: not needed (since only "initial" pe exists),

p-start-pe-option: not needed since save/resume feature is not available in this version. The PE is initialized on entry to transfer-enclosure, all multi-valued parameters adopting their initial/default values as in the agreed PE definition.

Class specific exit parameter usage

p-save-pe-option: not needed since save/resume feature is not available in this version.

P-CLOSE/ABORT-ENCLOSURE

not available in Basic Class.

The transfer-token will be owned initially by the p-user currently holding the enclosure-token (as in GVT/GPS). Any subsequent token passing service operation passes ownership of all the service tokens since no separate control over any token is provided in Basic Class.

2.3.7.2 Access Control Facilities

The access control facilities for the Basic Class are a simple subset of the powerful facilities described in the GVT.

Basic Class does not provide any subdivisioning of the CDS into parts with separate write access control and thus one and only one Write-Access-token is provided and is mapped onto the transfer-token. This write-access-token also covers CSS areas declared as being subject to token control.

The delivery control facility is available in Basic Class. Basic Class p-service provides no guarantee that information made available by a P(VT)-DELIVER will not be updated again at any time up to the passing of the write-access-token. Delivery and Receipt Acknowledgement in Basic Class is applicable to the whole CDS and to all controlled CSS areas.

2.3.7.3 Entry of Information to the CDS

Basic Class provides a single pointer to the CDS which indicates the next storage location at which data is to be entered. This location is known as the Active Location, coordinates (X,Y,Z). The value of the pointer is a combination of a coordinate value for each of the dimensions negotiated for the CDS. The initial value of the Pointer when a presentation environment is brought into use in transfer-enclosure in initialized state (the only case in Basic Class) is origin (1) on each existing dimension. The capabilities for explicit moves, i.e. changes in the values of these coordinates, are determined by the current definition of the PE. This pointer provides one mechanism for the entry or alteration of Attribute Values.

Ability to alter the Primary Values or Attribute Values of a cell or cells of the CDS is always conditional on owning the Write-Access-Token. Subject to this condition either p-user can, in principle, alter any attribute values.

Use of service elements in this section is thus subject to the possession of the write-access-token.

P(VT)-DATA

Available as defined in GVT.

Class specific parameter usage

The value of parameter p-cds-data for Basic Class is composed of some combination of sub-parameters as below, the values of which are described in terms of the Model of clause 2.2 and must conform to the current PE definition. These sub-parameters are subject to net-effect rule.

2.3.7.4 Position Control Sub-parameters

These sub-parameters of p-cds-data affect the value of the pointer, i.e. move the active location. Their use and the permissible values are dependent on the negotiated characteristics of the CDS in the PE.

NEW-LINE

x:=1 (home, origin of X), y:=y+1 (forward rel.move of 1). Invalid if Y is at bound of Y, or Y not defined. Valid if Y is defined, independent of X addressing attributes.
See parameters: p 20.4, 21, 22.

NEW-PAGE

x:=1 (home), y:=1 (home), z:=z+1 (forward rel.move of 1). Invalid if Z is at bound of Z, or Y or Z not defined. Valid if Z is defined, independent of X and Y addressing attributes.
See parameters: p 20.4, 21, 22, 23.

POINTER-ABSOLUTE 1 m n

x := 1, y := m, z := n.
If a dimension parameter omitted, this coordinate is unchanged.
Invalid if any of 1, m, n is outside bound of dimension or dimension is not defined, or if absolute addressing capability of the required kind is not in the negotiate capability.
See parameters: p 20.4, 21.1, 21.4, 22.1, 22.4, 23.1, 23.4.

POINTER-RELATIVE p q r

x := x + p, y := y + q, z := z + r (relative moves).
p, q, r can be +ve or -ve, -ve implies backwards move (see below).
If a dimension parameter omitted, this coordinate is unchanged.
Invalid if destination of move is outside bound of any dimension (including beyond origin), if a designated dimension is not defined or relative move is not available on it, or if a negative move specified and backward move is not available on the dimension.
See parameters: p 20.4, 21.1,.2,.3, 22.1,.2,.3, 23.1,.2,3.

POINTER-HOME X Y Z

x := 1 (home), y := 1 if Y defined, z :=1 if Z defined. If a dimension name omitted, this coordinate is unchanged. Invalid if for any designated dimension appropriate addressing capability is not "any".
See parameters: p 20.4, 21.4, 22.4, 23.4.

None of the position controls affect the value of any of the Objects in the CDS or empty cells, or the Attribute Values of any Cell.

2.3.7.5 Sub-parameters to Adjust Object Primary Values

The following sub-parameters affect the Primiary Values of one or more Atomic Objects. The action of these sub-parameters is always with reference to the current Active Location. The effect on the Active Location is as stated.

TEXT object-primary-value

Enter primary value into the atomic object in the cell at the current active location, and perform one implicit sequential X movement.
Invalid if entry attempted at limit or when any dimension coordinate is less than the lower window bound for the dimension.
See parameter p21 and sub-clause 2.2.3.1 and its sub-clauses.

Use of TEXT may bring into existence new cells -or arrays of cells or arrays of arrays of cells - if it follows a move beyond the maximum previously written value of a coordinate on an unbounded dimension. Sub-clause 2.2.3.1 gives the initial content of such newly generated cells. Use of TEXT does not alter any Attribute values.

Erase Operations

The following sub-parameters all Erase (set primary value to "nil") one or more Atomic Objects as given below. There can also be an effect on the Attributes of the corresponding Cells. This is dependent on the sub-parameter reset-attribute which can take symbolic value "yes" or "no". If "yes" all the Attribute values are also reset to their initial values as defined in the PE. If "no", attribute values are unchanged.

In all cases the extent of the action is as determined by the extent as given below. Cells (or arrays of ...) at coordinates lower than the lower window bound of a dimension will not be affected. New Cells will never be generated. There is no effect on the Pointer, i.e. the active location remains unchanged. The use of these sub-parameters, valid values of their parameters, and actual effects, are dependent on parameters of the PE definition.

ERASE-LINE extent reset-attribute

Allowed values of extent are:

Begin : erases from (1,y,z) to (x,y,z)

End : erases from (x,y,z) to (Xb or current length,y,z)

Whole : erases from (1,y,z) to (Xb or current length,y,z)

including the first and last position specified. x,y,z are the current coordinate values; y and/or z are ignored if the Y and/or Z dimension is not defined.

ERASE-PAGE extent reset-attribute

Allowed values of extent are:

Begin : erases all previous lines of current page and (1,y,z) to (x,y,z) on current line

End : erases from (x,y,z) to (Xb or current length, Yb or current length, z)

Whole : erases all of page, i.e. from (1,1,z) to (Xb or current length, Yb or current length, z)

including first and last cell specified, see note below.

For the purpose of this operation (only) the X-arrays (those parts of them used if X unbounded) are deemed to be aligned end-to-beginning, i.e. (Xb, y) is deemed to be followed by (1, y+1).

ERASE-BOOK extent reset-attribute

Allowed values of extent are:

Begin : erases all pages preceding current page, in current page all lines preceding current line, on current line all cells preceding and including current cell

End : erases all pages following current page, in current page all lines following current line, on current line all cells following and including current cell

Whole : erases complete CDS

including first and last cell specified.

For the purpose of this operation (only) (Xb, Yb) of one X-Y-array is deemed to be followed by (1,1) of the next. The note under ERASE-PAGE is applicable within each X-Y-array.

2.3.7.6 Sub-parameters for Character Attribute Values

The following sub-parameter causes designated attribute values of one or more cells of the CDS to be adjusted. The action can be, but is not necessarily, linked to the action of TEXT and/or the Position Control sub-parameters above; see attribute-extent below.

ATTRIBUTE attribute-id attribute-value attribute-extent

The values of the sub-parameters are constrained to be within relevant aspects of the Basic Class definition and the definition of the PE in use.

Sub-clause 3.4 gives the constraints applicable for the values of the parameters in the PE definition.

Allowed values of extent are:

- the whole CDS (the only valid value if specified attribute is a global attribute)
- from current active location over specified range of X
- any cell written by TEXT until this attribute mode is cancelled
- set of locations specified by explicit X Y Z values.

Attribute values in cells at coordinates below the lower window boundary on any dimension are not altered. New cells are never generated. The pointer is not affected.

2.3.7.7 Presentation to Other P-User

This aspect of Basic Class conforms to the general statements made in GVT with the following qualifying detail.

The following service elements are available as described in sub-clause 2.3.4:

P(VT)-REQUEST-TOKENS

P(VT)-GIVE-TOKENS

Class/enclosure specific usage aspects

Within a transfer-enclosure the write-access-token is mapped onto the transfer-token. Passing ownership of this using P(VT)-GIVE-TOKENS will also pass ownership of the enclosure-token and the p-terminate-token.

The following service elements are available in Basic Class as described in GVT, including parameter usage, except where qualified by Basic Class specific information, and as appropriate to the operating subset.

P(VT)-DELIVER

Class specific parameter usage

p-solicit-ack is available.
p-token-ident parameter is not used (in Basic Class only one write-access-token is used and also mediates issue of P(VT)-DELIVER request primitive and delivery is applicable to the whole CDS and all controlled CSS areas).

P(VT)-ACK-RECEIPT

Class specific parameter usage

p-token-ident parameter is not used (in Basic Class only one write-access-token is used and delivery and hence receipt acknowledgement is applicable to the whole CDS and all controlled CSS areas).

2.3.8 Control, Signalling and Status (CSS) Mechanisms

The facility for defining CSS mechanisms as described in GVT is available in Basic Class with the following class constraints applicable in this version.

The single write-access-token controls write access to the CDS and to any CSS data areas defined as subject to token control. Update of any such area defined as a "trigger" area will cause delivery of the update content of itself and any other token controlled areas (including CDS) and then release of the write-access-token to the peer p-user.

The user-defined CSS areas are restricted in this version of this Standard to be of uniform data type and realization is restricted to the general classes of "status", "signal" and "control". Further interpretation of such data is outside the scope of this version of this Standard.

For all "uncontrolled" CSS areas the p-service provides no facilities for enforcing any user conventions to avoid simultaneous updates or for coordinating the CCA content in such cases.

The default CSS areas as used in this version of Basic Class are covered in sub-clause 2.3.9.1.

2.3.9 Virtual Devices

The facilities for associating one or more virtual devices with the CCA and for associated CSS areas with each such device are available in Basic Class as described in GVT. Class specific restrictions are:

- since there is only one CDS component it is automatically associated with all the virtual devices.

The following service elements are available in Basic Class as described in GVT with the following qualifying detail.

P(VT)-CSS P(VT)-FREECSS	qualifying detail below applies to both cases.

Class specific usage detail

The p-css-info supplied for a designated CSS area must conform to the default or negotiated data format for the area in question. In this version of Basic Class these areas are of uniform type so the possible format of data parameter is similarly restricted.

2.3.9.1 Default CSS Areas

For each virtual device introduced under parameter 70 a default minimal CSS area is defined automatically. For Basic Class these default areas are as follows:

ident: the same as the name of the virtual device.

access control: "uncontrolled" subject to parameter 71.2.

trigger: "no" (does not act as a trigger).

format: 1 octet bit map, the bits are defined as follows:

bit 1	ON/OFF	(= 1 for ON)
bit 2	INTERRUPT	(= 1 for INTERRUPT)
bit 3	STATUS ALERT	(= 1 for ALERT)
bit 4	ALARM	(= 1 for ALARM)
bit 5-7	Reserved	

For each default CSS area the realization is "unspecified", i.e. the local mapping may associate any appropriate capability (and even more than one) with the logical facility; thus for example INTERRUPT may be associated with one or more of attention key, function key, light pen.

2.3.10 Service Exception Conditions

Basic Class provides no recovery mechanisms for continuing a presentation-connection after an exception condition. P-ABORT and P-DISCONNECT are applicable as described in GVT.

2.3.11 Diagnostic Information

This aspect of Basic Class conforms exactly to the statements in the GVT, (and hence by reference to GPS) and there are currently no specific statements to make in this Standard.

3 Protocol

3. PROTOCOL

3.1 Protocol Overview

The Basic Class Virtual Terminal Protocol conforms fully to the provisions of the ECMA Generic Data Presentation-Services Description and Protocol Definition (GPP), ECMA-86, and the ECMA Generic VT-Service and Protocol Description (GVT), ECMA 87.

3.1.1 Relationship Between Conceptual Service Interface Requests and Presentation Protocol Messages

Referring first to the Model View as depicted in the following diagram the p-users have agreed a common view of the Conceptual Communication Area.

A simple minimal set of primitive conceptual service interface commands 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
- write data
- read data.

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 that either of the p-users wish.

However this is tedious and unsatisfactory so that additional commands are introduced at the conceptual service interface level for convenience and additional protocol messages (such as ERASE-IN-LINE) are introduced for efficiency reasons. These two additions can be made independently as long as there is at least one way in which a new interface command can be expanded into the basic protocol set.

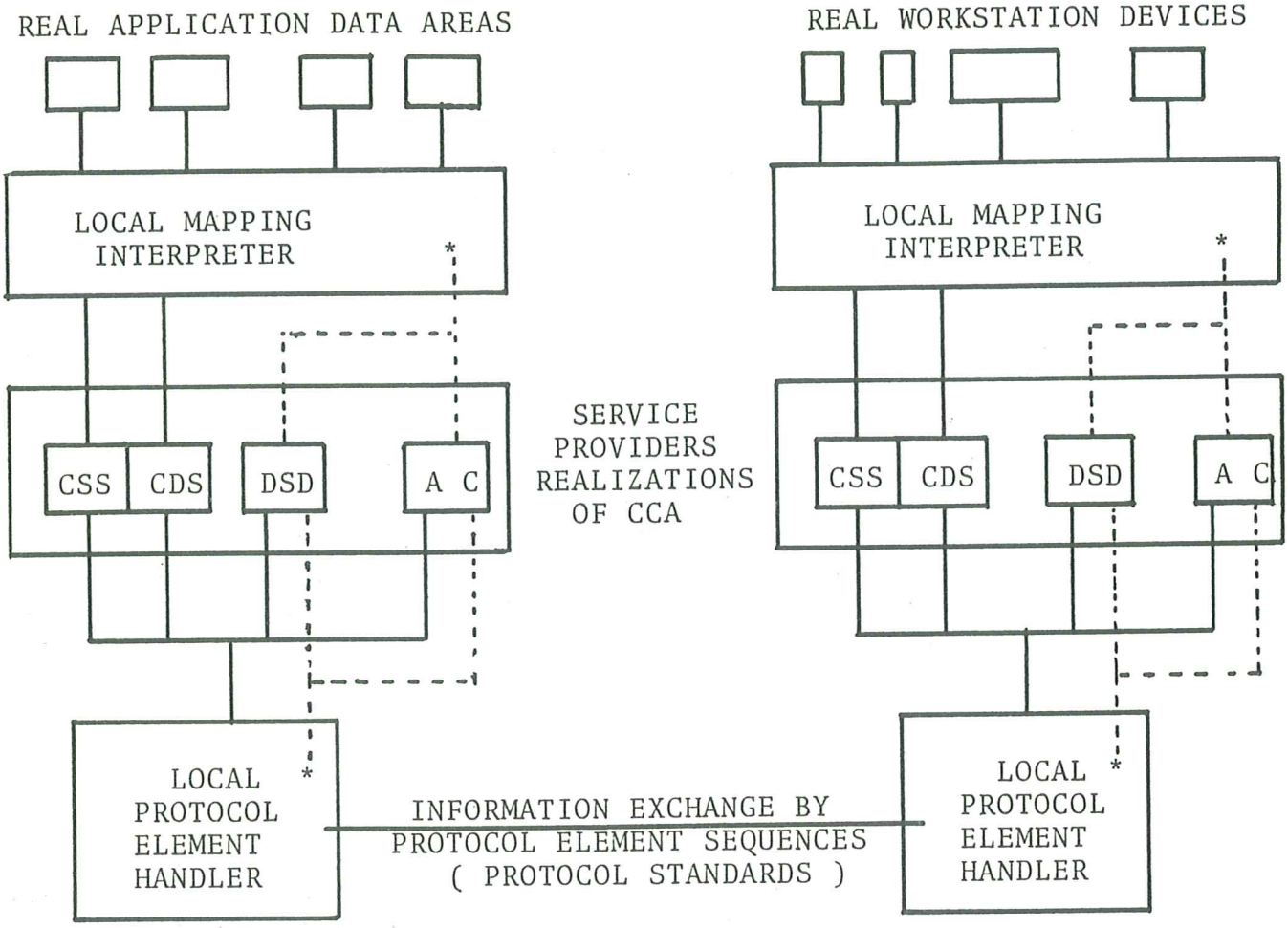


Figure 2. The Implementation View (Realization)

This Standard defines the protocol sequences and does not constrain the local mapping or the local protocol handler/generator so 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. This leaves the implementation 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".

3.1.2 Summary of Protocol Messages

A protocol message is the smallest quantity of information exchanged between peer entities which has a self-contained semantic significance. Although it is recognized that a message may have an internal structure (for instance it may imply the use of more than one Session Service primitive) for the purpose of communication between p-entities a message is considered to be an indivisible unit.

Request and response types of message are recognized. Some messages occur in isolation and do not require a particular

message as answer. Such messages are called request messages and each is said to be the (only) member of a type 1 protocol message group. Other messages require the receiving p-entity to respond with a particular message (sometimes the response must be generated immediately, sometimes other messages may be transmitted before the response is given). The first message of such a pair is a request message, the second is a response message. The message pair is referred to as a type 2 protocol message group.

A protocol message contains protocol control information (i.e. header and possibly one or more parameters) and may contain data.

In order to define the protocol it is necessary to define under which conditions (i.e. in which states) a certain message may be sent to the peer entity and which messages should be accepted from the peer entity. In all cases the actions and resulting state must be specified.

3.1.2.1 Summary List of Protocol Messages

PROTOCOL MESSAGES		MINIMUM SUBSETS VTB	MINIMUM SUBSETS GPP	TOKEN	NOTE
CONNECT protocol messages					
PP-CONNECT-REQ	CNQ	A			
PP-CONNECT-RESP	CNR	A			
RELEASE protocol messages					
PP-RELEASE-REQ	RLQ	A		Y	
PP-RELEASE-RESP	RLR	A		N	
DISCONNECT protocol message					
PP-DISC	DNQ	A		-	
CHANGE-ENCLOSURE protocol messages					
PP-CHANGE-ENCL-REQ	CEQ	D	C	Y	
PP-CHANGE-ENCL-RESP	CER	D	C	N	
PERFORM NEGOTIATION protocol messages					
PP-PERFORM-NEG-REQ	PNQ	C	C	Y	
PP-PERFORM-NEG-RESP	PNR	C	C	N	
NEGOTIATION ENCLOSURE protocol messages					
PP-NEG-INVITE	NIQ	D	C	Y	
PP-NEG-OFFER	NOQ	D	C	Y	
PP-NEG-ACCEPT	NAQ	D	C	Y	
PP-NEG-REJECT	NRQ	D	C	Y	

PROTOCOL MESSAGES		MINIMUM SUBSETS VTB	GPP	TOKEN	NOTE
TOKEN MANIPULATION protocol messages					
PP-GIVE-TOKENS	GTQ	A		Y	(3)
PP-REQUEST-TOKENS	RTQ	A		N	(3)
TRANSFER ENCLOSURE protocol messages					
PP-DATA	NDQ	A		Y	(1)
PP-CONTROL	NCQ	A		Y	(2)
PP-FREECONTROL	FCQ	A		-	(2)
PP-DELIVER	DLQ	A		Y	
PP-ACK-RECEIPT	ARQ	A		N	

Legend for this Summary

Subset VTB: indicates the lowest subset in the hierarchy of Basic Class subsets VTB-A, -B, -C, -D, in which the facility is available. A facility is never dropped in a higher subset.

Subset GPP: indicates whether a subset other than the lowest, (GP-A) is implied by the facility in isolation. It does not follow that a higher subset may not be required due to other considerations such as context switching between service, class, and/or subsets, or a requirement for negotiation of PE.

Subject to Token:

"No entry" in this column signifies that the token does not exist at this stage,
"Y" indicates possession of the token is required,
"N" indicates the token must not be currently possessed,
"-" indicates the position of the token is irrelevant.

The abbreviated names used in the summary correspond to those in the definitions of the messages in clause 3.2 and those used in the Formal Description in Appendix D. These short-form names are not intended for general use outside the Standard.

Notes for the Summary
(referenced by numbers in the final column)

1. PP-DATA is used only to carry CDS data. In this version CDS data is encoded according to ECMA-6, ECMA-35, ECMA-43, ECMA-48, subject to negotiated parameters of the PE.

2. PP-CONTROL and PP-FREECONTROL are both used to carry CSS data only. PP-CONTROL is used when the CSS area is controlled. PP-FREECONTROL is used when the CSS area is uncontrolled.

3. The entries here indicate the token requirements for the generation of the messages. Service requests are not similarly restricted, but a service request which would have a null effect will not result in generation of a message.

3.1.3 Conventions and Notation for Definition of Protocol Messages

Clause 3.2 provides a narrative definition of the protocol messages with indication of the ways in which they are used. Appendix D provides the formal description of the protocol.

Each message is defined by the following items:

- purpose
- list of parameters
- brief description of generation and reception conditions.

A detailed definition of the parameters is given only when they differ from the description of equivalent parameters of the service elements in Section II.

3.1.4 Mapping of Messages into Lower-Layer Service

This mapping is not considered in the narrative description in Clause 3.2 but is taken into account in Clause 3.3 which defines the encoding of the presentation protocol messages. Definition of the mapping is given in Clause 3.5. A consequence of the mapping is that some of the parameters in the messages will not need an explicit encoding in the Presentation Protocol Message Encodings since they are sent using specific parameters of the lower (session) layer services other than SS-user-data.

3.1.5 Relationship to GPP and GVT

Basic Class uses the protocol messages defined in GPP and/or GVT, where these are appropriate to its level of functionality. The parameters in such messages may not all be appropriate to Basic Class (e.g. the default value always applies or the parameter is wholly irrelevant); such parameters are omitted in this standard.

Explicit references to GPP and GVT are not, in general, given.

3.2 Protocol Message Definitions

This Clause defines the Protocol Messages which are applicable to Basic Class and indicates the way in which they are used. Details of encoding are given in Clause 3.3 and of mapping into Session Service primitives in Clause 3.5. Appendix D gives the full formal description of the protocol.

For each protocol message the purpose of the message is given, its content as a qualitative description of parameters con-

veyed by it, in some cases optional, and notes on the reason for generation by a p-entity, and on the actions taken if received by a p-entity; these descriptions are given as an aid to understanding the purpose of the element and to supplement but not replace the formal definition of the encoding of elements of protocol and contained parameters in Clause 3.3 or of the valid states and transitions of the presentation-protocol-machines (PPMs) in Appendix D. The message definition in this clause is independent of the way in which the message and its parameters may be conveyed between the two p-entities.

3.2.1 CONNECT protocol messages

PP-CONNECT-REQ (CNQ)

Purpose

To request the establishment of a presentation-connection and propose parameter values; solicits a CNR response to complete the protocol message group.

Contents

Parameters derived from the parameters shown as relevant to P-CONNECT request primitive as follows:

pp-protocol-definition: derived from service parameters p-service-ident, p-service-class-ident, p-service/class-version-idents, p-service/class-subset-ident, p-pe-profile as relevant for this class.

pp-session-subset: derived as for pp-protocol-definition.

pp-session-parameters: derived as for pp-protocol-definition, also possibly affected by other parameters of P-CONNECT; see Additional information.

pp-user-caller pp-user-called pp-fixed-pe-params pp-initial-pe-params pp-transp-data pp-special-conventions	These parameters are derived directly from the service parameters excluding any element mapped into the above protocol parameters.

Some parameters may not be present. See also Additional Information.

Generation

Generated when a p-user issues a valid P-CONNECT request primitive.

Reception

A valid CNQ will cause a P-CONNECT indication primitive from the PPM. If the p-service finds the parameters acceptable this indication will be passed to the p-user designated in the p-user identification information; otherwise the p-service will itself issue a CNR protocol message with appropriate diagnostic.

Additional information

The applicable parts of the service parameters p-fixed-pe-params and p-initial-pe-params are given under P-CONNECT service description.

PP-CONNECT-RESP (CNR)

Purpose

Response to CNQ to indicate success or failure of connection attempt.

Contents

Parameters derived from the parameters shown as relevant to P-CONNECT response primitive as follows:

pp-protocol-definition: derived from service parameters p-service/class-version-idents, p-service/class-subset-ident as relevant for this class.

pp-session-params: derived as for pp-protocol-definition, also possibly affected by other parameters of P-CONNECT; see Additional information.

pp-user-called
pp-fixed-pe-params
pp-initial-pe-params
pp-transp-data
pp-special-conventions

} Derived directly from the service parameters with exclusion of elements mapped as above into protocol parameters.

pp-diagnostic: derived from the service parameter p-diagnostic or supplied by the p-entity.

Some parameters may not be present, pp-diagnostic may be omitted if severity value is "success".

Generation

Generated by p-entity when the p-user has issued a P-CONNECT response primitive or when the p-entity is itself refusing to accept the connection.

Reception

A valid CNR will result in a P-CONNECT confirmation primitive to the p-user with appropriate p-diagnostic parameter.

Additional information

The applicable parts of service parameters p-fixed-pe-params and p-initial-pe-params are given under P-CONNECT service description.
If the diagnostic is present and the severity value indicates other than "success" the connection has not been established and the p-entities return to their previous state. Value "success with warning" is not used.

3.2.2 RELEASE protocol messages

PP-RELEASE-REQ (RLQ)

Purpose

To request the tidy termination of a p-connection; solicits a RLR response to complete the protocol message group.

Contents

Parameters derived from the parameters shown as relevant to P-RELEASE request primitive as follows:
pp-transp-data: derived directly from p-transp-data.

Generation

Generated if p-user issues P-RELEASE request primitive. Is sent following any outstanding protocol messages for protocol message groups initiated by the p-entity or needing completion by the p-entity.

Reception

A valid RLQ will result in delivery of any non-delivered changes to CDS and/or controlled CSS areas, followed by a P-RELEASE indication primitive to the p-user.

PP-RELEASE-RESP (RLR)

Purpose

Response to RLQ to indicate completion of release.

Contents

Parameters derived from the parameters shown as relevant to P-RELEASE response primitive as follows:
pp-transp-data: derived directly from p-transp-data,
pp-diagnostic: derived from the service parameter p-diagnostic or supplied by the p-entity. pp-diagnostic may be omitted if severity value is "success".

Generation

Generated by p-entity when the p-user issues a P-RELEASE response primitive. Is sent following any outstanding protocol messages for protocol message groups initiated by the p-entity or needing completion by the p-entity.

Reception

A valid RLR results in a P-RELEASE confirmation primitive to the p-user with appropriate p-diagnostic.

3.2.3 DISCONNECT protocol message

PP-DISC (DNQ)

Purpose

Used to force disconnection of the p-connection and the supporting s-connection when requested by a p-user or when found necessary by a p-entity. No response is solicited as part of this protocol message group.

Contents

Parameter pp-disc-reason containing either the p-transp-data from the P-DISCONNECT request primitive or a diagnostic code giving the p-entity's reason for forcing the disconnection.

Generation

Generated by p-entity when p-user issues P-DISCONNECT request primitive, or when a p-entity detects an exeption condition which forces abort of the p-connection.

Reception

A DNQ will cause the p-entity to abandon any activities in progress or queued and give P-DISCONNECT indication or P-ABORT indication primitive to the p-user according to the parameter value.

Additional information

Internal p-service abort and abort due to S-ABORT indication(s) are notified to the p-user(s) outside the scope of this Standard.

3.2.4 CHANGE ENCLOSURE protocol messages

PP-CHANGE-ENCL-REQ (CEQ)

Purpose

To request an orderly change of enclosure from the current one to a designated one; solicits a CER response to complete the protocol message group.

Contents

The parameters below are derived directly from the parameters shown as relevant to P-CHANGE-ENCLOSURE request primitive:
pp-dest-encl-type, (pp-exit-params), (pp-entry-params); see Additional information,
pp-transp-data.
Some parameters may not be present.

Generation

When the p-user issues a P-CHANGE-ENCLOSURE request primitive with parameters acceptable to the associated p-entity.

Reception

A valid CEQ will cause delivery of any non-delivered changes to the CDS and/or controlled CSS areas, followed by a P-CHANGE-ENCLOSURE indication primitive to the p-user. The parameters of the indication primitive may have been changed by the receiving presentation entity to reflect its capabilities. If the p-entity is unable to execute the request a CER will be generated without indication to the p-user.

Additional information

The parameter names pp-exit-params and pp-entry-params are generic names for one or more individual parameters; the actual parameters applicable are dependent on the value of pp-dest-encl-type and are given under the service descriptions.

PP-CHANGE-ENCL-RESP (CER)

Purpose

Response to CEQ to indicate success, partial success or failure of a request to make an orderly change from one enclosure to another.

Contents

The parameters below are derived directly from the parameters shown as relevant to P-CHANGE-ENCLOSURE response primitive:
pp-dest-encl-type, pp-transp-data; pp-diagnostic.
Some parameters may not be present; pp-diagnostic may be omitted if the severity value is "success".

Generation

When the p-entity itself rejects the CEQ or when the p-user issues a P-CHANGE-ENCLOSURE response primitive.

Reception

A valid CER causes a P-CHANGE-ENCLOSURE confirmation primitive to the p-user with appropriate p-diagnostic parameter.

3.2.5 GIVE TOKENS protocol message

PP-GIVE-TOKENS (GTQ)

Purpose

Used to pass the protocol token to the peer p-entity because a p-user has requested the passing of (all the) service tokens defined in the current phase/enclosure to the peer p-user, either explicitly or by use of a "trigger" CSS area. No response is solicited as part of this protocol message group.

Contents

There are no service derived parameters in GTQ in Basic Class, see Additional information.

Generation

Generated by p-entity when the p-user issues 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 due to previous P(VT)-DATA and/or P(VT)-CSS request primitives.

Reception

A valid GTQ causes a P(VT)-GIVE-TOKENS indication primitive to the p-user, with notification of any update information not already passed. The new possession of all service tokens will be recorded.

Additional information

Parameter pp-token-ident in GVT, is not used in Basic Class since all service tokens are always relinquished at the same time. GVT parameter pp-encl-specific also is not used.

3.2.6 REQUEST TOKENS protocol message

PP-REQUEST-TOKENS (RTQ)

Purpose

Used to request possession of the protocol token because a p-user has requested possession of (all the) service tokens defined in the current phase/enclosure. No response is solicited as part of this protocol message group. See Additional information.

Contents

There are no service derived parameters in RTQ in Basic Class, see Additional information.

Generation

Generated by p-entity qhen the p-user issues a P(VT)-REQUEST-TOKENS request primitive.

Reception

A valid RTQ causes a P(VT)-REQUEST-TOKENS indication primitive to the p-user.

Additional information

Parameter pp-token-ident in GVT, is not used in Basic Class since all service tokens must always be requested at the same time.

3.2.7 PERFORM NEGOTIATION protocol messages

PP-PERFORM-NEG-REQ (PNQ)

Purpose

To request the performance of a single-interaction negotiation enclosure; solicits a PNR response to complete the protocol message group.

Contents

The parameters below are derived directly from the parameters shown as relevant to P-PERFORM-NEGOTIATION request primitive:
pp-pe-specific-params; pp-transp-data.

Some parameters may not be present. See also Additional information.

Generation

When the p-user issues a P-PERFORM-NEGOTIATION request primitive with parameters acceptable to the associated p-entity.

Reception

A valid PNQ causes delivery of any non-delivered changes to the CDS and/or controlled CSS areas followed by a P-PERFORM-NEGOTIATION indication primitive to the p-user unless the p-entity is unable to execute the request in which case a PNR is generated.

Additional information

Parameter pp-pe-ident in GVT is not used. Values of parameters pp-draft-pe and pp-pe-specific-params are as given under P-PERFORM-NEGOTIATION description.

PP-PERFORM-NEG-RESP (PNR)

Purpose

Response to PNQ to indicate success or failure of the negotiation attempt.

Contents

The parameters below are derived directly from the parameters shown as relevant to P-PERFORM-NEGOTIATION response primitive:
pp-pe-specific-params; pp-transp-data; pp-diagnostic.

Some parameters may not be present; pp-diagnostic may be omitted if the severity value is "success". See also Additional information.

Generation

When the p-entity rejects a PNQ negotiation request or when the p-user issues a P-PERFORM-NEGOTIATION response primitive.

Reception

A PNR causes a P-PERFORM-NEGOTIATION confirmation primitive to the p-user with appropriate diagnostic.

Additional information

If the diagnostic severity indicates other than "success" the previous state is restored. Severity value "success with warning" is not relevant. See also Additional information to PNQ.

3.2.8 NEGOTIATION ENCLOSURE protocol messages

PP-NEG-INVITE (NIQ)

Purpose

To request an offer for a set of parameters, no response is explicitly solicited as part of this protocol message group.

Contents

The parameters below are derived directly from the parameters shown as relevant to P-NEG-INVITE request primitive: pp-param-ident-list; pp-transp-data.
Some parameters may not be present.

Generation

When the p-user issues a P-NEG-INVITE request primitive with p-invite-param-list compatible with the current state of the Draft PE.

Reception

A valid NIQ causes a P-NEG-INVITE indication primitive to the p-user.

PP-NEG-OFFER (NOQ)

Purpose

To offer one or more values or ranges of values for a set of parameters. No response is explicitly solicited as part of this protocol message group.

Contents

The parameters below are derived directly from the parameters shown as relevant to P-NEG-OFFER request primitive: pp-param-list; pp-transp-data.
Some parameters may not be present.

Generation

When the p-user issues a P-NEG-OFFER request primitive with p-offer-param-list compatible with the current state of the Draft PE.

Reception

A valid NOQ causes a P-NEG-OFFER indication primitive to the p-user.

PP-NEG-ACCEPT (NAQ)

Purpose

To indicate acceptance of the value or value range of a set of parameters. No response is explicitly solicited as part of this protocol message group.

Contents

The parameters below are derived directly from the parameters shown as relevant to P-NEG-ACCEPT request primitive: pp-param-list; pp-transp-data.
Some parameters may not be present.

Generation

When the p-user issues a P-NEG-ACCEPT request primitive with p-accept-param-list compatible with the current state of the Draft PE.

Reception

A valid NAQ causes a P-NEG-ACCEPT indication primitive to the p-user.

PP-NEG-REJECT (NRQ)

Purpose

To indicate rejection of one or more parameters in a previous offer. No response is explicitly solicited as part of this protocol message group.

Contents

The parameters below are derived directly from the parameters shown as relevant to P-NEG-REJECT request primitive: pp-param-ident-list; pp-transp-data.
Some parameters may not be present.

Generation

When the p-user issues a P-NEG-REJECT request primitive.

Reception

A valid NRQ causes a P-NEG-REJECT indication primitive to the p-user.

3.2.9 TRANSFER ENCLOSURE protocol messages

PP-DATA (NDQ)

Purpose

Used to convey CDS update information. No response is explicitly solicited as part of this protocol message group.

Contents

One or more items of CDS update information derived from one or more P(VT)-DATA request primitives. See sub-clause 3.3.3 and clause 3.4.

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, will cause the sending of one or more NDQ (and NCQ) protocol messages if there are any pending CDS and/or CSS (controlled) updates, followed by the appropriate message.

Reception

A valid NDQ 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 defined in clause 3.4.

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 explicitly 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: pp-css-area-ident, pp-css-info; this pair of parameter types may occur more than once.

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 explicitly 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:
pp-css-area-ident, pp-css-info; this pair of parameter types may occur more than once.

Generation

Generated by p-entity when the p-user issues a P(VT)-FREECSS request primitive. Sequence with other FCQ 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. Such acknowledgement forms a separate protocol message group.

Contents

The parameter below is derived directly from the parameter shown as relevant to P(VT)-DELIVER request primitive; pp-solicit-ack.

Generation

Generated if p-user issues P(VT)-DELIVER request primitive, following (logically) transmission of any NDQ and/or NCQ protocol messages due to previous P(VT)-DATA or P(VT)-CSS request primitives. It is never generated autonomously by the p-entity itself.

Reception

A valid DLQ will cause the p-entity to give a P(VT)-DELIVER indication primitive to the p-user, with notification of any update information not already passed.

Additional information

In Basic class deliver applies to the whole CDS and all "controlled" CSS areas.
Parameter pp-token-ident included in DLQ in GVT is not used in Basic Class.

PP-ACK-RECEIPT (ARQ)

Purpose

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

Contents

None; see Additional information.

Generation

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

Reception

A valid ARQ will cause the p-entity to give a P(VT)-ACK-RECEIPT indication primitive to the p-user and release any protocol interlock applied pending this reception.

Additional information

In Basic Class the deliver facility and hence any interlock applied applies to the whole CDS and all "controlled" CSS areas.
Parameter pp-token-ident included in ARQ in GVT is not used in Basic Class.

3.3 Protocol Encoding

Bit numbering convention

The bits of an octet are identified as f1, f2, f3, ..., f7, f8, where f1 is the left-most and most-significant bit of the octet. A similar convention applies to bit strings longer than one octet, with the right-most and least-significant bit becoming f16, f24, etc.

Note 3

"f" is used in this convention to distinguish this convention from that used in character coding tables, etc., where the numbering is in the reverse direction, i.e. b1 is the least-significant bit, and extension, e.g. from 7 to 8 bit codes, is by addition to the left.

3.3.1 Protocol message Structure

The protocol messages defined in clause 3.2 are transferred between the peer p-entities using the services of the session connection in different ways.

Each protocol message is composed of two parts:

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

Some messages are uniquely mapped into a session-service primitive, i.e. form the only possible content of such primitive; the presentation header can be omitted since this is implied by the session service used for the transfer of the p-message. Whether this is actually done is defined in clause 3.5. Frequently for such messages some part of the content as defined in Clause 3.2 will be mapped into the specific parameters of the session service primitive and such parts will, in general, not be explicitly encoded in the presentation protocol data unit which is mapped into the SS-user-data field (if any) of the primitive.

Some other messages are completely mapped, including both header and all content, into the SS-user-data field of S-DATA or S-TYPED-DATA service.

Full definition of the mapping between presentation protocol messages and the session service primitives is given in Clause 3.5.

3.3.2 Presentation Header

The single-octet presentation header is used as an 8-bit Message Type with values from 0 to 255, encoded in binary.

3.3.2.1 Principle of Allocation of Message Type

The Generic Presentation Protocol Standard (GPP) gives a rule for allocation of message type codes. The value range is divided up as follows:

0	Reserved, not used as a Type Code.
1 to 63	Allocated for presentation protocol control messages defined within the GPP. Values are assigned from 1 upwards. The messages used by Basic Class VTP with the type codes are listed in sub-clause 3.3.5.
64 to 255	Allocated as available for definition in specific protocol standards without requirement for uniqueness between such standards.

The GVT defines some further message types and Basic Class VTP conforms to this definition. Further message types specific to Basic Class use the message type range from 128 upwards as defined in GVT.

3.3.3 Presentation Message Content

The structure of this is dependent on the message type. In many cases it will consist of one or more parameters encoded as defined in 3.3.4; in other cases there will be an implicit encoding of the information, dependent on the message type and defined for each case. The final (unstructured) part of the content may not have an explicitly encoded length but be implicitly bounded by the SSDU into which the message is mapped.

Clause 3.4 gives the encoding of the contents of NDQ messages, see sub-clause 3.2.9.

3.3.4 Parameter Encoding

This uses a subset of the "TLV" technique as defined above.

3.3.4.1 Type/Length/Value (TLV) Encoding

The TLV technique is a method for coding of an information unit such as a "parameter".
As used in this Standard each such unit is made up of three parts:

- type (first field)
- length (second field)
- value (third field)

Type field - one octet

f1 = 0
f2-f8 = type value: 1 to 127, 0 reserved.

Length field - 1 or 2 octets

f1 = 0 : length L specified on 7 bits (f2 - f8)
= 1 : length L specified on 15 bits (f2 - f16)

f2-f8/f16 : binary number L of octets of the value field.

Value field - variable length

L octets of data, internal structure and coding is dependent on Type, and may be a nested TLV (compound value).

3.3.4.2 Parameter Value Representation

When specifying the type of encoding used in the value field of parameters the following convention is used:

C: character string: Any size (unless limits are explicitly defined). Character encoding is according to ECMA 7-bit coded character set, see ECMA-6. Each character is mapped on bits f2-f8 of an octet, with bit f1 = 0. The allowed characters are those in columns 2 to 7 of the International Reference Version, excluding that in position 7/15. Particular parameters may apply further constraints on the range of characters.

N: numeric: Unsigned binary integer with four possible sizes: 8, 16, 24, or 32 bits (L=1, 2, 3 or 4).

S: symbolic: Unsigned binary integer the values of which encode specific meanings. Size is 8 bits (L=1).

M: bit-map: Bit string in which each bit encodes a specific meaning. Size is 8 or 16 bits (L=1 or 2). Any bit the role of which is not defined must be set to 0. For bits representing specified options, the bit is set to 1 if the option is requested, to 0 otherwise.

T: transparent: Coding and meanings explicitly defined or p-service is transparent to it. Conveyed as a binary octet string.

A: aggregate value: Value is a structure of fields explicitly defined for the particular parameter; the fields may, but will not necessarily, conform to the above classifications, in particular the sizes may be smaller. Conveyed as a binary octet string.

P: compound parameter: Value is made up of a nested structure of parameters, see 3.3.4.4.

Value truncation: For economy reasons it is recommended to use the minimum variable length string for expressing values:
- character strings should not contain unnecessary spaces,
- for numeric values, all unnecessary octets from left containing only zero bits should be removed,
- for symbolic values, the octet may be removed (length becomes 0) if the symbolic value is that corresponding to binary value 0,
- for bit-map values, all unnecessary octets from right containing only zero bits should be removed (extension of bit-map values will be to the right), thus the length may become 0,
- for transparent values, no specific truncation recommendation can be given; particular definitions may define a truncation algorithm.

3.3.4.3 Parameter Encoding

For each parameter of the protocol, the following information is given:

- Parameter type value: This is unique among the parameters which can occur at the particular level of the "parameter tree".
- Value representation type: C, N, S, M, T, A, or P.
- Maximum length of value field: "-" means unlimited (in the general definition).
- Encoding of all possible values: for S fields.
- Significance of all bits: for M fields.
- All possible component parameters: for P values.
- Detailed structure: for A values.

For parameters which are mapped directly into the parameters of the session service this information is not usually required where the relevant information is given in the session standard, but supplementary information may be needed and will be given in Clause 3.5.

Default Values

The existence of a default value which will be assumed if the parameter is not explicitly specified, depends partly on the value type for the parameter and partly on the specific definition of the parameter:

- For value types S and M there will generally be a default value, equal to 0.
- For value types C and T there is no default.
- For value type N there is generally no default value, but a default may be defined for some such parameters.

3.3.4.4 Nested Parameters

In general the value field of a TLV-encoded parameter can itself be a collection of TLV-encoded parameters. Such a parameter is known as a compound parameter, and its value as a compound value; value type designation is "P". Each of the contained parameters can also be a compound parameter, to any required depth of nesting. Where a parameter value is not one or more TLV parameters it is a simple parameter of the nested parameter structure. The value of such a parameter can be either an elementary value (value types C, N, S, M, T) or an aggregate value (value type A); in the "A" case there is an explicit substructure defined explicitly for that parameter.

This Standard includes a number of compound parameters. Parameters which are always contained within a particular "outer" parameter have type values unique among the total set of parameters which can occur within the value field of the outer parameter but which may overlap the type values of other parameters which can never occur at this "node".

3.3.4.5 Range and List Parameters

To enable the declaration, in connection with a particular parameter type, of a range of values, i.e. with a low limit and a high limit, or of a list of specific discrete values, two special parameter type values are defined as follows:

Range Parameter : type 32

List Parameter : type 31

These always have a Compound Value. The components can be a further Range or List parameter or can be some other parameter type; in this second case these component parameters must be of the same type.

In the case of a Range parameter the components will be two in number; the first gives the low limit and the second the high limit.

In the case of a List parameter any number of components can be present, each giving a specific entry value for the list; this value can be a range, a list, or a single compound or simple parameter.

The occurrence of a parameter or compound parameter in a List or Range is only permitted if this usage is explicitly allowed in the parameter description. Subject to this, any parameter type may appear in a List parameter, only Numeric or Numeric/Symbolic parameters may occur in a Range parameter.

If a List parameter specifies a set of parameters one or more of which must be selected, (i.e. an offer), then the sequence of the parameters in the offered set

indicates the preference for the alternative selection of the sending service user. The first element is preferred most, followed by preference for the second element, and so on.

3.3.5 Message Type Codes

3.3.5.1 Protocol Messages Used from the GPP

The Table below lists the Message Type Codes defined in GPP which are used in this Standard.

	PROTOCOL MESSAGE		TYPE CODE
	PP-CONNECT-REQ	CNQ	1
	PP-CONNECT-RESP	CNR	2
	PP-RELEASE-REQ	RLQ	3
	PP-RELEASE-RESP	RLR	4
*	PP-DISC	DNQ	(5)
	PP-PERFORM-NEG-REQ	PNQ	8
	PP-PERFORM-NEG-RESP	PNR	9
	PP-CHANGE-ENCL-REQ	CEQ	10
	PP-CHANGE-ENCL-RESP	CER	11
	PP-NEG-INVITE	NIQ	24
	PP-NEG-OFFER	NOQ	25
	PP-NEG-ACCEPT	NAQ	26
	PP-NEG-REJECT	NRQ	27

* *This type value is reserved but is not used since the message is mapped into a session service primitive with restricted field for presentation data.*

3.3.5.2 Protocol Messages Used from the GVT

The Table below lists the Message Type Codes defined in GVT which are used 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

3.3.5.3 Protocol Messages Specific to Basic Class

In this version there are no protocol messages specific to Basic Class.

3.3.6 Parameter Encoding Details

The parameters used in this Standard are encoded according to the principles in sub-clause 3.3.4 using, firstly, the definitions given in GPP and secondly, the definitions in GVT. The following tables list the parameters to increasing degrees of detail and specificity to this Basic Class Standard. "Forward references" are given in the tables to the next level of detail where the table does not give the "terminal" definition. Numbers in parentheses refer to additional information following the tables.

The rule for ordering of parameters encoded within protocol messages or within compound parameters is implied by the order in which the parameters are given in parameter lists or tables, unless some other specific statement is made. A receiving p-entity can, but is not obliged to, consider as invalid message which violates such a rule.

3.3.6.1 Usage of Parameters Defined in GPP

The Table below lists the parameters defined in GPP which are relevant to Basic Class. Specific encodings amending or extending those given in GPP, are given in some cases, in other there are forward references to the next level of definition.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
pp-diagnostic	1	-	A	aggregate value, see 3.3.7.1
pp-transp-data	2	48(1)	T	
pp-protocol-definition	3	-	A	aggregate value, see 3.3.7.2
pp-fixed-pe-params	4	-	P	compound value, see 3.3.6.1.2
pp-initial-pe-params	5	-	P	compound value, see 3.3.6.1.1
pp-dest-encl-type	9	-	S	0:not used 1:"nul", 2:"transfer" 3:"negotiation"; other values reserved
pp-pe-specific-params	10	-	P	compound value, see 3.3.6.1.3

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
pp-pe-action	15	1	S	0:not used 1:"switch", 2:"restore"; other values reserved.
pp-param-ident-list	16	-	P	compound value, see 3.3.6.1.4
pp-param-list	17	-	P	compound value, see 3.3.6.1.5
pp-special-conventions	22	-	T	user-defined
pp-special-info	23	-	T	user-defined

Additional information

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.

3.3.6.1.1 Content of pp-initial-pe-params

In this version the only component parameter of pp-initial-pe-params is pp-protocol-definition, see sub-clause 3.3.7.2, which gives information on the permitted content of this when included in this parameter.

3.3.6.1.2 Content of pp-fixed-pe-params

In this version the only possible component parameter of pp-fixed-pe-params is pp-special-conventions with user-defined content.

3.3.6.1.3 Content of pp-pe-specific-params

This can include component parameters as given in 3.3.6.1.1 for pp-initial-pe-params, which will then be the first component parameter. It can also contain parameters from the list given in sub-clause 3.3.6.3; multiple occurence is possible as described in sub-clause 3.3.6.1.5. It can also contain parameter pp-special-info with user-defined content.

3.3.6.1.4 Content of pp-param-ident-list

Component parameters of pp-param-ident-list are TLV'd items, each with a "type" as defined in sub-clause 3.3.6.3, but with no "value" field (length = 0). Each type can occur at most once and the order must be as in 3.3.6.3.

3.3.6.1.5 Content of pp-param-list

Component parameters of pp-param-list are TLV'd parameters as defined in sub-clause 3.3.6.3. Occurrence of a parameter of a particular type as a range or list is permissible depending on the type and on the protocol message. A list of lists or ranges may be appropriate in some cases. The order in which type values appear must be as in 3.3.6.3.

The rule for multiple occurrence is as follows:

PROTOCOL MESSAGE		OCCURRENCE IN RANGE OR LIST
PNQ	NOQ	Any Parameter in 3.3.6.3
PNR	NAQ	Any Parameter in 3.3.6.3 which is indicated as Multi-valued in the table in sub-clause 2.3.6.4, unless explicitly stated otherwise.

All occurrences of a multi-occurrence parameter in a PNR or NAQ message must be in one List.

The occurrences of a single-occurrence parameter in a PNQ or NOQ message may be specified, within the parameter in which its occurrences must occur according to this encoding structure, in one or more Lists and, if applicable for the parameter type, one or more Ranges.

If one List is offered, one of the occurrences in the List should be selected.

More than one List for a certain parameter may only occur if the parameter is a compound parameter and each different List makes an independent offer for a part of the compound parameter. This form is not allowed for Numeric or Numeric/Symbolic parameters.

Numeric and Numeric/Symbolic parameters may be offered as one List and one or more Ranges. The List and Range parameters for one Numeric or Numeric/Symbolic parameter must be adjacent within the parameter containing this offer.

3.3.6.2 Usage of Parameters Defined in GVT

The Table lists the parameters defined in GVT which are relevant to Basic Class. Specific encodings, amending or extending those given in GVT, 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-solicit-ack	60	1	M	f1 = 1 "yes" = 0 "no" Other bits reserved
* pp-css-area-ident	59	-	C	user-defined
pp-css-info	58	-	according to CSS definition; for default CSS area see 3.3.6.3.11	

* *The default CSS-area for a virtual device is identified by the identifier of the virtual device itself as the value of this parameter.*

3.3.6.3 Specific Parameters for Basic Class

This sub-clause defines the encoding for the remaining parameters in this Standard, which are specific to Basic Class (i.e. not defined in GVT; however some other VTS classes may use some of them, with the same or different encodings and rules for use).

The definition is with reference to the list of parameters in the Basic Class Presentation Environment given in sub-clause 2.3.6.4, noting that parameters p1 through p11.1 are covered by sub-clause 3.3.6.1 (by pp-protocol-definition).

The encoding takes some account of more general requirements in GVT although these are not covered in the current version of GVT.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
erasure-capability (p12)	64	1	M	f1 = 1 "yes" = 0 "no" Other bits = 0
encoding-methods (p13.1, 13.2)	65	2	A	Aggregate value, see 3.3.6.3.1
CDS-characteristics (p20.4)	66	2	P	Compound value, see 3.3.6.3.2
X-dimension-descr (p21)	67	5	P	Compound value, see 3.3.6.3.3
Y-dimension-descr (p22)	68	5	P	Compound value, see 3.3.6.3.3

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
Z-dimension-descr (p23)	69	5	P	Compound value, see 3.3.6.3.3
character-attributes (p31)	70	-	P	Compound value, see 3.3.6.3.4
CSS-descriptor (p60)	71	-	P	Compound value, see 3.3.6.3.5
virtual-device-descriptor (p70)	72	-	P	Compound value, see 3.3.6.3.6

Parameters p60 and p70 are indicated as multiple occurrence parameters in sub-clause 2.3.6.4. However each parameter p60 or p70 together with its unique and mandatory identifying identifier parameter (p60.1 or p71 resp) should be regarded as a single occurrence parameter. This implies:

- multiple occurrences of this parameter in a PNR or NAQ message should not be placed in a List,
- if an offer is made for a CSS area or Virtual Device which necessitates the occurrence of p60 or p70 respectively in a List parameter then all entries in one List must have the same name. (Each such list within pp-param-list offers one CSS area or one Virtual Device).

3.3.6.3.1 Content of "encoding-methods"

The compound value has components as in the table below. These sub-parameters are used only as components of the outer parameter as above.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
controls-encoding (p13.1)	1	1	S	0 :"void", no value designated, 1 :"embedded-7", (ECMA-48, 7-bit) 2 :"embedded-8", (ECMA-48, 8-bit) 3-15: reserved, 16-255: user-defined.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
graphics-encoding (p13.2)	2	1	S	0 :"void", no value designated, 1 :"ISO 7-bits" 2 :"ISO 8-bits"
graphics-set-capability (p13.2.1)	3	1	N	0 : not used, >0 : number of graphic sets
graphics-set-name (p13.2.1.1)	4	-	C	registered graphics code set, user defined name, or "void" (length=0)

3.3.6.3.2 Content of "CDS-characteristics"

The compound value has a single component in Basic Class as in the table below. This sub-parameter is used only as a component of the outer parameter as above.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
number-of-dimensions (p20.4)	4	1	N	values 1,2,3 are valid

3.3.6.3.3 Content of Dimension Descriptors p21, p22, p23

The compound value has components as in the table below (references are given for the p21.n case). These sub-parameters are used only as components of the outer parameters as above.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
bound (p21.1)	1	-	S/N	0 : "unbounded" >0 : value of bound
max-forward-rel-move (21.2)	2	-	S/N	0 : "any", >0 : permitted move plus one (i.e. 1 implies none)
max-backward-rel-move (21.3)	3	-	S/N	as above.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
absolute-move	4	1	S	0 : "none" 1 : "higher", 2 : "any". Other values reserved
window-width	5	-	S/N	0 : "any", >0 : width plus one (i.e. 1 implies zero width)

A component parameter is omitted if a value is not requird to be designated.

3.3.6.3.4 Content of "character-attributes" p31

The compound value has components as in the table below. These sub-parameters are used only as components of the outer parameters as above.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
repertoire-capability (p31.1)	1	1	N	0 : not used, >0 : number of repertoires
repertoire-descr	2	-	P	compound value, see below
emphasis-capability (p31.2)	5	1	N	0 : not used, >0 : number of emphasis values
emphasis-name (p31.2.1)	6	-	C	name
colour-capability (p31.3)	7	1	N	0 : not used, >0 : number of colour values
colour-name (p31.3.1)	8	-	C	name
background-colour-capability (p31.4)	9	1	N	0 : not used, >0 : number of background col values
background-colour-name (p31.4.1)	10	-	C	name

A component parameter is omitted if a value is not required to be designated.

The components of repertoire-descr are as follows:

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
repertoire-name (p31.1.1)	2	-	C	registered repertoire or user name or "void"
font-capability (p31.1.1.1)	3	1	N	0 : not used, >0 : number of fonts
font-name (p31.1.1.1.1)	4	-	C	registered font or user name or "void"

A component parameter is omitted if a value is not required to be designated.

An occurrence of one of the parameters p31.1, p31.2, p31.3, p31.4 and p31.1.1.1 may be followed by a list parameter containing occurrences of the parameter of which the number of occurrences is being controlled. In a PNR or NAQ message this subsidiary parameter may occur as many times as required up to the value of the relevant capability parameter, forming an ordered list (defaults have to be added if the list is shorter than the capability). In addition to that, in PNQ and NOQ messages, the list may be longer than the capability; this should be interpreted as indicating that the receiver may select any combination of "n" elements out of the set of offered values, where "n" is the capability.

In line with the rules specified in 3.3.6.1.5 it is permissible to specify in pp-param-list one list of p31 making an offer for p31.2 and p31.2.1 and another list making an offer for p31.3 and p31.3.1.

It is not allowed to offer a capability parameter (as range or list) followed by an offer (via a list) for the subsidiary parameter.

3.3.6.3.5 Content of "CSS-descriptor" p60

The compound value has components as in the table below. These sub-parameters are used only as components of the outer parameter as above.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
css-descr-ident (p60.1)	1	-	C	user-defined
css-modes (p60.2, 60.3)	2	1	M	f1=0:p60.2 is unspecified f1=1,f2=0:p60.2 "no" f1=1,f2=1: "yes" f3=0:p60.3 is unspecified f3=1,f4=0:p60.3 "no" f3=1,f4=1: "yes" Other bits reserved
css-data-format (p60.4)	3	2	A	Aggregate value, see below

A component parameter is omitted if a value is not required to be designated.

Aggregate value of css-data-format is encoded as follows:

First octet: p60.4.1 : S type :
0 : "M", 1 : "C", 2 : "N", 3 : "S",
4 : "T".
Other values reserved.

Second octet:p60.4.2 : N type :
0 not used,
>0 : field length in octets.

3.3.6.3.6 Content of "virtual-device-descriptor" p70

The compound value has components as in the table below. These sub-parameters are used only as components of the outer parameter as above.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
virtual-device-ident-&-modes (p71)	1	-	P	Compound value, see 3.3.6.3.7
repertoire-realization (p74)	4	-	P	Compound value, see 3.3.6.3.8
* emphasis-realization (p75)	5	-	C	name or "void"
* foreground-colour-realization (p76)	6	-	C	name or "void"

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
* background-colour-realization (p77)	7	-	C	name or "void"
layout-control (p78)	8	-	P	Compound value, see 3.3.6.3.9
CSS-areas (p79)	9	-	A	Aggregate value, see 3.3.6.3.10

This compound parameter will occur once for each virtual device.

* *These parameters can each occur as many times as required up to the value of the relevant capability parameter, see 3.3.6.3.4, forming an ordered list. "Void" is encoded as a non-existant value, i.e. L=0 for the parameter. Refer to ECMA-48 for the encoding of the emphasis and colour names.*

3.3.6.3.7 Value Definition for "virtual-device-ident-&-Modes"

The component sub-parameters are defined as in the table below. These sub-parameters are only used as components of the outer parameter as above.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
virtual-device-ident (p71)	1	-	C	user defined
virtual-device-modes (p71.1, p71.2)	2	1	M	f1=0: p71.1 is unspecified f1=1,f2=0:p71.1"off" f1=1,f2=1: "on" f3=0: p71.2 is unspecified f3=1,f4=0:p71.2"no" f3=1,f4=1: "yes" Other bits reserved

3.3.6.3.8 Value Definition for "repertoire-realization"

Component parameters are as follows; these sub-parameters are used only as components of the outer parameter as above:

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
repertoire-realization (p74.1)	1	-	C	registered-repertoire or user-name or "void"
font-realization (p74.1.1)	2	-	C	registered-font or user-name or "void"

3.3.6.3.9 Value Definition for "layout-control"

The component sub-parameters of the compound value are defined in the table below. These sub-parameters are only used within "layout-control" sub-parameter, see 3.3.6.3.6.

PARAMETER NAME	TYPE	LENGTH (MAX)	VALUE TYPE	VALUE ENCODING
X-layout (p78.1)	1	-	S/N	Special value type see below
Y-layout (p78.2)	2	-	S/N	Special value type see below

The special value type is one or more octets used as a binary number with meanings as follows:

0 : "unbounded"
>0 : value of explicit layout control

3.3.6.3.10 Value Definition for "CSS-area"

The aggregate value is used as follows:

First octet : S type : p79.2
0 : "void", i.e. undeclared,
1 : "status",
2 : "device control",
3 : "signal",
Other values reserved.

Remaining octets : C type : p79.1
User-defined, see p60.1, 3.3.6.3.5.

The sub-parameter "CSS-area" occurs in parameter p70 (see 3.3.6.3.6) once for each CSS area required for the virtual device.

3.3.6.3.11 Value Definition for pp-css-info for Default CSS Area

This is given in sub-clause 2.3.9.1.

3.3.7 Special Parameter Encodings

In some cases, because of limited availability of space for a parameter, e.g. due to a restriction in the session service parameters, or the desire to keep the overheads to a minimum for a frequently used message, the parameters are not encoded using the standard scheme as defined in 3.3.4.

3.3.7.1 Encoding of Diagnostic Parameter

In most messages this is encoded using a standard TLV format and the value of Type is given in the table above. The value is an "aggregate" parameter as given below.

The value field for pp-diagnostic is a structure defined as follows using BNF notation:

<diagnostic value>::=<severity><reason> [<DS>]

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

The terminal elements are encoded as follows:

<severity>
4-bit binary integer; f1-f4 of first octet of <diagnostic value>. Values are defined below. The default value assumed when the diagnostic parameter is omitted is 0 (success).

<reason>
12-bit binary integer: f5-f8 of first octet of <diagnostic value> followed by f1-f8 of second octet. Values are defined below together with the messages in which the values can appear. The default value assumed when the diagnostic parameter is omitted is 0 (no reason given).

<DS type>
3-bit binary integer; f1-f3 of first octet of <DS>. Values 0,1,2 are used, other values are reserved.
<DS value length>
5-bit binary integer; f4-f8 of first octet of <DS>.
<DS value>
Dependent on <DS type> as follows:

DS type = 0 : character string, maximum length 31 characters, conforming to "C-type" parameter values.

DS type = 1 : one octet containing a parameter type (see 3.3.6) or 2 or more octets containing nested parameter types, in the case of parameters recursively encoded within parameters.

DS type = 2 : one octet containing a message type (see 3.3.5).

3.3.7.2 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 and GVT:

pp-service-ident: single octet, S type value :
2 : "VTS" is applicable for Basic Class.
0 : field is "void"; see Additional information.

pp-service/class-version-idents: single octet, M type value :
f1=1 (version 1), is applicable bit/value for this version of Basic Class; (f2 to f7 and f8 are not relevant to this version but may be used for selection of later versions when such standards exist).
All bits = 0; field is "void", see Additional information.

pp-service/class-subset-ident: single octet, M type value :
f1 : =1 basic class subset BC-A,
f2 - f4 : similarly subsets BC-B to BC-D, (multiple occurrence permissible).
f5 : reserved for future standardization.
f6 : = 1 "delivery-control",
= 0 "no delivery-control".
f7 : = 1 field value "void", other bits set to 0.
f8 : = 0 in this version (extension of subset field is not relevant).

pp-service-class-ident: single octet, S type value :
1 : applicable value to select "basic class". (GVT, gives the currently defined set of values).
0 : field is "void"; see Additional information.
This field is omitted if it is "void" and the pp-pe-profile field is not required.

pp-pe-profile: variable length, A type value :
encoding of this field is defined in sub-clause 3.3.7.5. This field is omitted if not required.

Additional information

pp-protocol-definition is used in two ways in this version of Basic Class:

a) encoded at the outer level of parameter encoding in CNQ and CNR messages, and thus having "fixed" status. In this instance of pp-protocol-definition 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 Basic Class and of the related standards these parameters of the p-connection must be "fixed"). The other fields defined above are optional.

b) encoded as the first parameter of pp-initial-pe-params of pp-pe-specific-params. In such an occurrence of pp-protocol-definition the fields given in (a) as mandatory must have "void" values. The other fields defined above can take values as defined above.

3.3.7.3 Encoding of Parameter pp-pe-profile

This is a variable length aggregate value type parameter used in pp-protocol-definition. Its length is implicitly determined by the length of pp-protocol-definition.

The first octet is used to identify the profile from the set of standard Basic Class profiles, the remaining octets give the mandatory profile parameters.

First octet: this is used as follows:
f1 : reserved, set to 0,
f2-f8 : S-type 7-bit value:
0 : not used, 1 : "P1", 2 : "P2",
3 : "P3", 4 : "P4",
values 4-63 reserved for future standardization,
values above 63 available for user definition.

The use of the remainder of the parameter is dependent on the profile identified in the first octet.

Profile P1: second octet is used as follows: M type value:

f1 : = 1 if r1 is 2 octets, = 0 if one octet.
Other bits are reserved.

The remainder is a one or two octet field as determined by f1 above, and is used as a binary number as follows:

0 : not used,
>0 : profile parameter r1 (X-bound).

Profile P2: second octet is used as follows:
M type value:

f1 : = 1 if r1 is 2 octets,
f2 : = 1 if r2 is 2 octets,
f5 : profile parameter r3: = 1 "yes"
= 0 "no",
Other bits are reserved.

The remainder is used as two fields, each one or two octets long. The first contains profile parameter r1 and the second contains r2, the lengths of each being indicated by f1 and f2 respectively in the second octet as above. Usage of the binary values is as for r1 in profile P1 above.

Profile P3: encoding is as for Profile P2 above.

Profile P4: second octet is used as follows:
M type value:

f1 : = 1 if r1 is 2 octets,
f2 : = 1 if r2 is 2 octets,
f3 : = 1 if r3 is 2 octets,
Other bits are reserved.

The remainder is used as three fields, each one or two octets long. The first contains profile parameter r1, the second contains r2 and the third contains r3, the lengths of each being indicated by f1, f2 and f3 respectively in the second octet as above. Usage of the binary values is as for r1 in profile P1 above.

The maximum length of the above value field is thus 8 octets. Further profiles defined in a later version or in some external register may exceed this length; this may affect the available length of p-transp-data in CNQ message.

3.3.8 Values of Severity in Diagnostic Parameters

Values are defined as follows:

0 : success (default if diagnostic not present)
1 : success with warning
2 : reserved
3 : failure

Other values are reserved.

3.3.9 Values of Reason in Diagnostic Parameters

Values are defined as follows: (the number at the extreme right indicates the severity codes associated with the reason).

0	: no reason provided (default if diagnostic not present)	0
1	: non-standard reason	1,2,3
2	: unset parameter value	3
3	: illegal parameter value	3
4	: unsupported parameter value	3
5	: illegally duplicated parameter	3
6	: illegal parameter type	3
7	: unsupported parameter type	3
8	: illegal message type	3
9	: unsupported message type	3
10	: failed due to collision	3
11	: invalid p-user response parameter	3
12	: protocol sequence violation	3
13	: presentation environment incomplete	3
17	: negotiation-enclosure not supported, not entered	2
18	: negotiation not supported	3
19	: illegally duplicated parameter value	3
20	: conflicting parameter value	3
21	: illegal parameter sequence	3

3.4 Inter-Relationship Between Service Parameters and Protocol Message Elements

3.4.1 Introduction

This Clause defines the permissible contents of the PP-DATA-REQ (NDQ) messages, in terms of protocol message elements.

In this version the protocol message elements are derived from Standards ECMA-6, ECMA-35, ECMA-43 and ECMA-48. In some cases a combination of the units in these referenced standards is necessary to represent a discrete functional element.

There are several types of constraints on the use and sequencing of the protocol message elements and units available in Basic Class.

The purpose of this Clause is to define the constraints, as imposed by this Standard, between values of negotiable Service Parameters and allowable protocol Elements, Units and Sequences. Such constraints allow implementors to choose the subset of elements, units and sequences which will be implemented, on the understanding that, to conform to this Standard, the implementation must:

a) honour the constraints (when generating protocol messages) applicable to the specific set of agreed service parameters in use at the time (the PE),

b) refuse to accept any value of any parameter which entitles the peer entity to generate any element/sequence outside the range of the implementation.

While it is not, in general, necessary, in the interests of implementation freedom, to define a precise mapping between conceptual service elements (let alone actual implemented interface elements) and protocol sequences, this standard chooses, in view of the extensive range of parameterization being offered, to specify some general mapping constraints to allow for reasonable subsetting when supporting real devices with limited capabilities.

The constraints specified in this Clause are additional to those implied in the definition of the protocol itself and its defined subsets. The latter set are implied in the protocol definition and reflected directly in that certain service facilities are only available in certain states. The primary components of the "state" are the applicable subset, the current phase or enclosure and the possession of relevant token(s). The Table in sub-clause 3.1.2 summarizes some of the constraints of this type; the remainder of this sub-clause should be used with this in mind.

This version of this Standard itself defines the structure of all protocol elements with the single exception of the encoding of the contents of PP-DATA protocol message. For this, existing encodings and escapes as defined in ECMA-6, ECMA-35, ECMA-43 and ECMA-48 are used. The following Table shows which facilities of these existing encodings are allowed in PP-DATA protocol messages. Any facilities of the referenced standards which are not listed in the table are not allowed. Explanatory notes follow the table, indicated by (n) in the table.

3.4.2 Protocol Elements Within PP-DATA Protocol Messages

Table Use of ECMA-6, ECMA-35, ECMA-43, ECMA-48 and of the International Register within PP-DATA-REQ Protocol Messages in Basic Class.

In the parts of the table below the elements are categorized in a similar manner to that in the referenced standards.

All elements given in the referenced standards which are not explicitly listed in the table are not allowed in PP-DATA messages in Basic Class.

Key to Table:

Y	: element is allowed
negotiable	: allowed subject to some negotiable parameter
restricted	: allowed subject to some restriction of the full definition.

In those situations where the encoded representation of the transferred graphic characters is not known (no transfer syntax) this version of this Standard defines that the

individual characters of the graphic repertoire are individual elements of graphic sets of 94 elements which can be represented in accordance with ECMA-35.

Graphics and CO Control Functions from ECMA-6, ECMA-43.			
ACRONYM	NAME	USED	NOTES
ggg	encoded representation of designated and invoked graphic characters	Y	(1) (2)
ESC	ESCAPE	Y	(4)
SI	SHIFT-IN	negotiable	
SO	SHIFT-OUT	negotiable	
NUL	NULL	Y	(5)
SP	SPACE	Y	(2)
CR	CARRIAGE RETURN	negotiable	(2)
LF	LINE FEED	restricted	(2)
FF	FORM FEED	restricted	(2)
BS	BACKSPACE	Y	(3)

Editor Functions for Erase from ECMA-48			
ACRONYM	NAME	USED	NOTES
ECH	ERASE CHARACTER	negotiable	
ED	ERASE IN DISPLAY	negotiable	
EL	ERASE IN LINE	negotiable	

Introducers and Shifts from ECMA-35 and ECMA-48			
ACRONYM	NAME	USED	NOTES
CSI	CONTROL SEQUENCE INTRODUCER	Y	(4)
SS2	SINGLE-SHIFT TWO	negotiable	(6)
SS3	SINGLE-SHIFT THREE	negotiable	(6)
LSO	LOCKING-SHIFT ZERO	negotiable	(6)
LS1	LOCKING-SHIFT ONE	negotiable	(6)
LS2	LOCKING-SHIFT TWO	negotiable	(6)
LS3	LOCKING-SHIFT THREE	negotiable	(6)
LS1R	LOCKING-SHIFT ONE RIGHT	negotiable	(6)
LS2R	LOCKING-SHIFT TWO RIGHT	negotiable	(6)
LS3R	LOCKING-SHIFT THREE RIGHT	negotiable	(6)

Format Effectors from ECMA-48			
ACRONYM	NAME	USED	NOTES
HPA	HORIZONTAL POSITION ABSOLUTE	negotiable	
HPB	HORIZONTAL POSITION BACKWARD	negotiable	
HPR	HORIZONTAL POSITION RELATIVE	negotiable	
VPA	VERTICAL POSITION ABSOLUTE	negotiable	
VPB	VERTICAL POSITION BACKWARD	negotiable	
VPR	VERTICAL POSITION RELATIVE	negotiable	
PPA	PAGE POSITION ABSOLUTE	negotiable	
PPB	PAGE POSITION BACKWARD	negotiable	
PPR	PAGE POSITION RELATIVE	negotiable	
HVP	HORIZONTAL AND VERTICAL POSITION	negotiable	
SGR	SELECT GRAPHIC RENDITION	negotiable	

Miscellaneous from ECMA-48			
ACRONYM	NAME	USED	NOTES
REP	REPEAT	Y	

Notes to Table

1) Excluding the C0 and C1 controls there are 96 characters available in 7-bit (or 2 x 96 using SI and SO) and 2 x 96 in 8-bit. Of the two special cases, SPACE (SP) is always permitted (listed separately in the table) and DELETE (DEL) is excluded. The remaining 94 characters are available.

2) Although their functions are subsumed in those of other control functions used, certain C0 controls are allowed with certain qualifications as follows:

   SP (SPACE) : Always permitted (see also 1) above)

   CR : Permitted in isolation for certain service parameter values

   LF, FF : Only permitted when immediately preceded by CR and also subject to certain service parameter values. The special interpretation of LF as "New Line" is specifically excluded.

3) BS is permitted as an alternative to HPB (1). It is not used as a means of encoding a composite character by superimposition.

4) ESC and CSI are permitted only when required as part of the representation of control characters which are listed in this table, or for the designation of character repertoires.

5) NUL is permitted only in the ISO-defined meaning of "no operation" at the transfer syntax level; it does not enter "nil" or any other value into the CDS or affect the pointer in any way.

6) The availability of this element depends on the parameters "graphic-information-encoding" and "graphic-code-set-capability". The dependency on the first is as described in ECMA-35, the dependency on the second is described in sub-clause 3.4.3.10.

3.4.3 Detailed Protocol Constraints from Service Parameter Values

This sub-clause gives further detail of the constraints applied by the values of negotiated service parameters on the protocol messages and parameters.

3.4.3.1 Parameters 1, 2, 3, 11, Service, Class, Version, Operating Subset and Profile

These are specified (and/or "fixed") at connection establishment or on entry to negotiation. They have a "global effect" on the protocol elements, sequences and encoding which may be used. The overall set of elements/units/sequences is service- , class- and version-dependent. The encodings may differ in subsequent versions of this Standard. The subset definition indicates which of the overall set for the version concerned are usable in the subset in question. Any profile used in the setting up of a presentation environment implies the total of all the constraints that the particular parameter values it contains imply, until such time as a value may be re-negotiated or modified as a result of an agreed change to an antecedent parameter in the directed graph for the class/version concerned.

3.4.3.2 Parameter 11.1 Delivery Control

This parameter is not explicitly visible at the service level but it is implicitly linked to the operating subset. It has value "no" for subset VTB-A and "yes" for subsets VTB-B, VTB-C and VTB-D.

3.4.3.3 Parameter 12 Erasure Capability

Value "yes" allows use of the erasure controls in the Table, i.e. ECH, EL, ED, depending on the agreed dimensionality and adressing capabilities of the CDS. Value "none" implies that none of these is permitted.

3.4.3.4 Parameter 13 Control and Character Encoding

This version only allows the embedded control encoding according to ECMA-48 in PP-DATA messages with the standard 7-bit or 8-bit coded representation, and completely user-defined alternatives. Thus these parameters define generally the encoding technique and do not impose specific restrictions.

3.4.3.5 Parameters 21 X-dimension, 22 Y-dimension, 23 Z-dimension

The X-dimension always exists. Whether or not the Y- and Z-dimensions are Defined determines whether certain addressing controls (format effectors) may be used in PP-DATA.

Specifically:

Definition of Y allows CR.LF independently of p21.4, 22.2

Definition of Z allows CR.FF independently of 21.4, 22.4 and 23.2

If Y is not defined, VPA VPR VPB HVP CR.LF CR.FF and ED are illegal

If Z is not defined, PPA PPR PPB and CR.FF are illegal.

3.4.3.6 Parameters 21.2, 22.2, 23.2 Max forward relative moves

These respectively limit the value of the argument in the controls HPR(n), VPR(n), PPR(n). A maximum of 0 for any dimension prohibits the use of the control for that dimension altogether. The default for these parameters when the dimension exists is "any" since forwards relative movement is assumed normal.

3.4.3.7 Parameters 21.3, 22.3, 23.3 Max backwards rel moves

These respectively limit the value of the argument in the controls HPB(n), VPB(n), PPB(n). A maximum of 0 for any dimension prohibits the use of the control for that dimension altogether. The default for these parameters when the dimension exists is 0; use of backwards moves must therefore be negotiated explicitly or by profile. The use of CR is also controlled in that if a value of X = 1 cannot be legally reached using HPR(n) then CR is not legal in isolation; see also sub-clause 3.4.3.5.

3.4.3.8 Parameters 21.4, 22.4, 23.4 Absolute addressing

Values "none" for these respectively prohibit HPA, VPA, PPA. Value "higher" respectively permit any move which logically could be encoded as HPR(1).VPR(m).PPR(n) being alternatively encoded as HPA(x+1).VPA(y+m).PPA(z+n) where x,y,z are the current coordinate values, and/or combined into HVP.

Value "any" permits any -PR(n) or -PB(n) to be alternatively encoded as -PA(c ± n) and/or combinations into HVP (c is any of x,y,z as before). The use of CR is also controlled in that if a value of X = 1 cannot be legally reached using HPA(n) then CR is not legal in isolation; see also sub-clause 3.4.3.5.

If the logical relative moves are prohibited by 21.2, 21.3, etc. then the alternative absolute encoding must be used if permitted. If neither is permitted the move is illegal.

3.4.3.9 Parameters 21.1 X-bound, 22.1 Y-bound, 23.1 Z-bound

If a finite bound is specified this constrains the argument of the -PA in the appropriate dimension and the appropriate argument in the multidimensional control HVP.

While an absolute move to outside the bound can be declared illegal statically, relative moves with an argument value less than the declared bound in the dimension may still, when combined with the current coordinate value, result in an illegal coordinate value. Therefore no constraint is imposed on the negotiable value for maximum forward or maximum backwards moves by the existence of a bound. Use of a relative move will be validated in context and if the result coordinate value is outside the agreed bound the move is at that point diagnosed as illegal.

3.4.3.10 Parameters 31.1, 31.2, 31.3, 31.4 Repertoire, Emphasis, Foreground and Background Colour Capability

The numeric parameters specified here determine the variability of the character attributes across the CDS. The number determines the minimum number of "logical bits" of attribute information necessary to cover the negotiated repertoire. The SGR control, however, provides a separate argument encoding for each value of each attribute except 31.1. The particular values of SGR argument are determined by the above parameters and parameters 74 repertoire realization, 75 emphasis realization and 76 and 77 colour mappings.

If the graphic set capability (p13.2.1) is:

1 : Then G0 only is assumed and no shifts are allowed,

2 : Then G0, G1 and G2 are assumed. Character sets are designated to G0 and G1 unless their designation to G2 is required in accordance with ISO 6937, in which case SS2 is available. Otherwise SS2 and (always) SS3 are not allowed. In the 7-bit case G0

and G1 are accessed using SO and SI. In the 8-bit case it is assumed that G0 is loaded as if by LS0 and G1 as if by LS1R. Loading is assumed to be effected when G0 and G1 are designated. SO and SI are not used in 8-bit.

3 : Then G0 and G1 are assumed as for 2 above, and G2 is assumed. In the 7-bit case SI, SO, SS2 and LS2 may be used, in the 8-bit case LS0, LS2, SS2, LS1R and LS2R may be used, in accordance with ECMA-35.

≥ 4 : Then G0 and G1 are assumed as above and G2 and G3 are assumed. In the 7-bit case SI, SO, SS2 and LS2 may be used, in the 8-bit case LS0, LS1, LS2, SS2, LS1R and LS2R may be used in accordance with ECMA-35.

3.5 Session Service Mapping

The Basic Class Virtual Terminal Protocol relies on the Services offered by the Session Service as described in ECMA-75. This version of Basic Class uses version 1 of this Standard. In the event of a new version of ECMA-75 offering extended facilities it is possible that a later version of Basic Class may take advantage of these extensions. It is likely that the mappings defined in this version would remain a negotiable option (to the extent that the session conformance requirements permitted this).

3.5.1 Summary of Usage of Session Services

The following list shows the session service request/response primitive events which are issued for the various protocol messages defined in clause 3.2.

CNQ	S-CONNECT request
CNR	S-CONNECT response
RLQ	S-RELEASE request
RLR	S-RELEASE response
DNQ	S-DISCONNECT request
CEQ	S-DATA
CER	S-TYPED-DATA
GTQ	S-TOKENS-GIVE
RTQ	S-REQUEST-TOKENS
PNQ	S-DATA
PNR	S-TYPED-DATA
NIQ	S-DATA
NOQ	S-DATA
NAQ	S-DATA

NRQ	S-DATA
NDQ	S-DATA
NCQ	S-DATA
FCQ	S-TYPED-DATA
DLQ	S-SYNC request
ARQ	S-SYNC response

3.5.2 Connection Mapping

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

3.5.3 Session Connection Establishment

The establishment of a session-connection between the two presentation-entities is necessary before any presentation protocol activity can take place.

The establishment phase of the presentation-connection is mapped into the establishment facility of Session Layer, the PP-CONNECT-REQ (CNQ) protocol message being mapped (in part) into the SS-user-data parameter of the S-CONNECT request primitive and indication primitive, and the PP-CONNECT-RESP (CNR) protocol message being mapped (in part) into the SS-user-data parameter of the S-CONNECT response and confirmation primitives.

The following parameters of CNQ and CNR are not mapped into SS-user-data but are mapped into other parameters of S-CONNECT primitives as in ECMA-75.

- pp-user-caller into "initiator address"
- pp-user-called into "acceptor address"
- pp-session-subset into "subset choice": this version of Basic Class uses only subset E.
- pp-session-params into "subset parameters": for this version of basic class "data-token" is relevant and value is "initiator". "terminate-token" value is "initiator".

The result parameter of S-CONNECT response primitive is derived from pp-diagnostic parameter of CNR; if severity value is "failure" result is "rejected", otherwise it is "accepted".

The message header and all remaining parameters of CNQ and CNR are encoded transparently into SS-user-data as defined in 3.3.

3.5.4 Session Connection Termination

The various presentation protocol messages used for termination of presentation-connection (arising from the three forms of presentation service element) are mapped into the facilities of the session service as follows.

3.5.4.1 Mapping of P-RELEASE

The PP-RELEASE-REQ (RLQ) protocol message is mapped into the SS-user-data parameter of the S-RELEASE request and indication primitives, and the PP-RELEASE-RESP (RLR) protocol message is mapped into the SS-user-data parameter of the S-RELEASE response and confirmation primitives.

The result parameter of S-RELEASE response primitive is derived from pp-diagnostic parameter of RLR; if severity value is "failure" result is "negative", otherwise it is "affirmative".

The message header and all remaining parameters of RLQ and RLR are encoded transparently into SS-user-data as defined in 3.3.

3.5.4.2 Mapping of P-DISCONNECT

The PP-DISC (DNQ) protocol message is mapped into the SS-user-data parameter of the S-DISCONNECT request and indication primitives. Presentation protocol message type code is not used for DNQ as no other protocol message is mapped into the above session primitives.

The pp-disc-reason parameter of DNQ is encoded transparently into SS-user-data as defined in 3.3.7.3.

3.5.4.3 Effect of S-ABORT

Abort of p-connection due to S-ABORT is notified to the p-user(s) outside the scope of this Standard.

3.5.5 Other Session Service Facilities

In this Standard other session facilities are used as follows.

3.5.5.1 Session Token Control Facilities

The session token control facilities are used for the exchange of certain presentation protocol messages as follows.

PP-GIVE-TOKENS (GTQ)

This protocol message is mapped into the S-TOKENS-GIVE request and indication primitives with the tokens-given parameter having value "data token" and "terminate-token" and "synchronize-token" always.

PP-REQUEST-TOKENS (RTQ)

This protocol message is mapped into the S-PLEASE request and indication primitives with the tokens-requested parameter having value "data token" and "terminate-token" and "synchronize-token" always. The SS-user-data parameter is not used.

3.5.5.2 Session Minor Synchronization

S-SYNC request/indication/response/confirmation primitives are used for the transfer of pp-protocol messages PP-DELIVER DLQ and PP-ACK-RECEIPT ARQ.

The non-urgent type of S-SYNC request/indication is used to carry the DLQ resulting from a P(VT)-DELIVER request which does not solicit an acknowledgement.

The urgent type will be used to carry the DLQ if the P(VT)-DELIVER request does solicit acknowledgement. In this case S-TYPED-DATA can still be sent while the S-SYNC response/confirmation is awaited whereas S-DATA cannot. This constraint reflects the constraints implied by use of P(VT)-DELIVER with acknowledgement in view of the mapping shown in the next two sub-clauses.

In neither case is the parameter pp-solicit-ack of DLQ message encoded explicitly, the "type" parameter of the S-SYNC request/indication primitives being used to convey this information.

S-SYNC response/confirmation is used to carry the ARQ resulting from a P(VT)-ACK-RECEIPT request.

Since S-SYNC is not used for any other purpose and the Virtual Terminal Service is defined so that there can never be more than one P(VT)-DELIVER request with a pending solicited acknowledgement the VT service is able to supply the parameters required in the S-SYNC response/confirmation primitives which are therefore not visible to the VT service users.

3.5.5.3 Session Token-controlled Data Transfer Facility

S-DATA request/indication primitives are used for the transfer of a number of pp-protocol messages as listed below. Each complete protocol message is mapped into one SS-user-data parameter of one S-DATA request/indication primitive. The message type code is always present. Since blocking is not performed in the session subset used each p-protocol message will be sent as one session protocol message.

Applicable protocol messages:

CEQ PNQ NIQ NOQ NAQ NRQ

NDQ NCQ

The length of a NDQ message may exceed the maximum length of the SSDU. In such a situation the stream of protocol message elements may be stored in a number of subsequent NDQ messages. The protocol message element stream may be split on all octet boundaries except:

1 : between octets which form together one escape sequence,

2 : between octets which form together one Control String,

3 : between octets which must be together to be legal (e.g. CR LF),

4 : between octets which together form one graphic character.

3.5.5.4 Session Un-controlled Data Transfer Facility

S-TYPED-DATA request/indication primitives are used for the transfer of a number of pp-protocol messages as listed below. Each complete protocol message is mapped into one SS-user-data parameter of one S-TYPED-DATA request/indication primitive. The message type code is always present. Since blocking is not performed in the session subset used each p-protocol message will be sent as one session protocol message.

Applicable protocol messages:

CER PNR FCQ

4 Conformance

4. CONFORMANCE

4.1 Conformance Requirements

4.1.1 General

This clause defines the conformance requirements for the Basic Class Virtual Terminal Protocol defined in Section III of the Standard.

There is no conformance requirement for the Basic Class Virtual Terminal Service described in Section II of the Standard.

Only the externally visible and externally testable criteria are defined.

4.1.2 Equipment

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

4.1.3 Peer Equipment

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

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

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

4.1.4 Protocol Subsets

The equipment shall implement one or more of the protocol subsets defined in Clause 2.3.2.1 and shall nominate those subsets for which conformance is claimed.

4.1.5 Additional Virtual Terminal Protocols

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

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

4.1.6 Requirements

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

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

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

The set of "correct sequences" depends on parameters agreed during connection or negotiation.

Conformance with a particular subset requires that the following sequences are treated as correct sequences:

- sequences which are "correct sequences" when all parameters adopt their default value,
- if other parameters have been agreed (which in itself is not required for conformance) the "correct sequences" of the agreed set of parameters.

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.

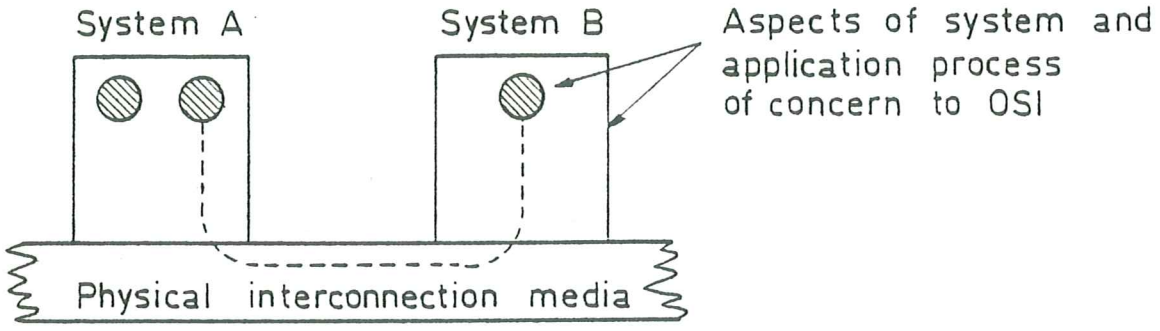


Fig. A1 - General Schematic Diagram Illustrating the Basic Elements of Open Systems Interconnection

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.

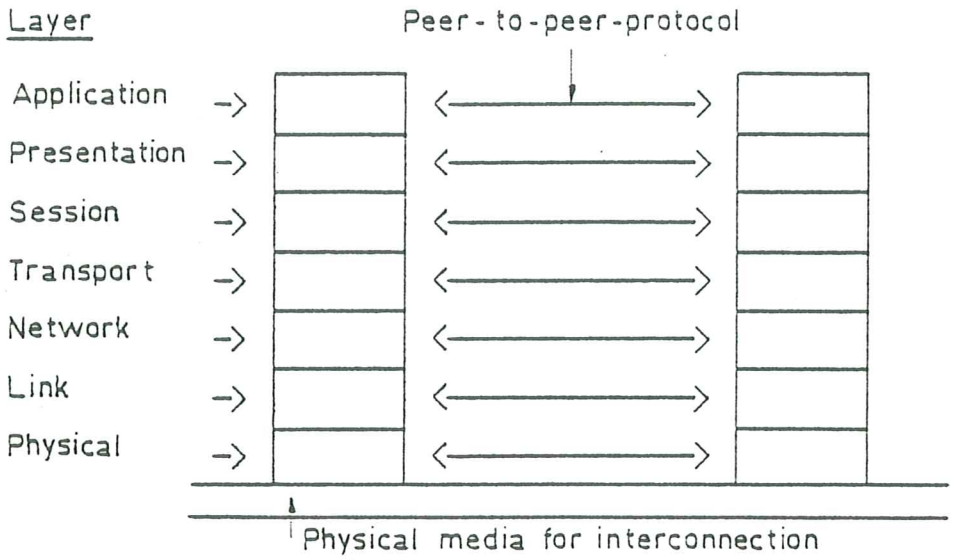


Fig. A2 - The Seven-Layer Reference Model and Peer-to-peer Protocol

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, which is defined in ISO 7498.
- 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 acronyms is provided in B.3.

B.2 Glossary of Terms

This glossary only covers technical words and phrases used either in a way which may not be self evident from the normal English meaning of the words, or with a particular restricted meaning within the context of this Standard. The explanations here are specific to Basic Class and may differ from the explanations of the same words and phrases in the GVT because of this. In many cases a reference is given to the text of the Standard where a more detailed explanation may be found. In other cases a reference to the text of the GVT may be necessary for a fuller explanation.

Access Control, AC (2.3.6.2, GVT)

Used to refer to the mechanism which regulates the two p-users' respective rights to access (read, update, erase) information in the shared conceptual communication area (CCA). Also used to denote the component of the CCA where the information relevant to such regulation is placed.

Access Token (2.2.1, GVT)

Used when the access control regulates a particular set of actions of the two p-users on a strict two-way-alternate basis. Ownership of the token is passed between the p-users and only the one currently owning the token has the right to perform the associated actions. The term is qualified as necessary to cover specific sets of actions (e.g. write-access-token).

Active Location (2.3.6.3)

The cell of the conceptual data store (CDS) which is currently addressed by the pointer. This is the cell into which the next character written will be placed.

Application Program (1.1.1)

Used in a restricted sense when an application program is one of the two interacting presentation service users. In normal usage of the Virtual Terminal Service (VTS) one user will be an application program and the other a terminal operator though both could be application programs.

Atomic Object (2.2.2, GVT)

The smallest separately addressable unit of information in the CDS with a distinct semantic significance within the context of the VTS. For the basic class this is a character. The object may be structured from a number of parts with different semantic significance (e.g. the primary value and character attributes).

Basic Class (1.2, GVT)

The Virtual Terminal Service (VTS) is described, and the Virtual Terminal Protocol (VTP) defined, for each of a series of (logical) device classes. Each class has significantly different fundamental capabilities. One of these is the Basic Class which is primarily intended for line-orientated or page-orientated character imaging devices used in an unstructured sequential manner.

Cell (2.2.4.2)

The smallest individually addressable element of the shared conceptual data store (CDS). It has a single, unique address (setting of the pointer) and the capacity to store one and only one Atomic Object.

Character Attribute (2.2.4.2)

Since the atomic object of the basic class is conceptually a character it is appropriate to term any attribute within the basic class atomic object a character attribute.

Conceptual Communication Area, CCA, (2.1, GVT)

This is the component of the Virtual Terminal Model which is the conceptual repository for all the information, data or control, being shared by the p-users using the Virtual Terminal Service. Within VTS there is no other means of communication between the p-users. The CCA consists of a conceptual data store, a data structure description, a control, signal and status information area, and an access control store.

Conceptual Data Store (2.1, GVT)

That part of the CCA which is the principal means of normal data communication between the VTS p-users. It has a structure agreed (by profile or negotiation) between the p-users and this structure is recorded in the DSD component of the CCA. It consists of a one, two, or three dimensional array of cells each

capable of holding one character and such character attributes as have been agreed for the particular VTS connection.

Conceptual Image (2.1, GVT)

The bulk of the information being shared between the two p-users can be interpreted as a conceptual image. Information in the CDS interpreted in conjunction with the structure information in the DSD and the control information in the AC will yield a conceptual image appropriate for (one or more) virtual devices as currently seen by each p-user.

Conceptual Interface Command Interpreter, CICI, (2.1, GVT)

The two p-users exchange information by placing it in and extracting it from the CCA. It is assumed that they do this by issuing commands across a conceptual service interface. The CICI interprets such commands and changes or delivers the content of the CCA accordingly. Since this Standard is not intended to define an actual interface the inclusion of the adjective conceptual in these terms is essential.

Conceptual Service Interface, CSI, (2.3.1)

As an aid to describing the services offered by the VTS and the purpose of each of the protocol elements, a means by which the p-users could make their requests known to the service is introduced. Since this is in no way the subject or standardization it is termed a conceptual service interface.

CSS Area (2.1)

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 visualise 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 (2.3.2, 2.3.7.1)

The Defined PE which is in use in transfer-enclosure. In Basic Class it is the "initial" and only PE.

Data Structure Definition, DSD (2.2.3, GVT)

The part of the CCA which records the mutually agreed control information, the Defined PE, which includes the structure of the other components (CDS and CSS). In particular it records

the dimensionality and boundedness of the CDS as well as the substructuring of the atomic object into primary value and character attributes, and the existence and essential (to the VT) characteristics of the control, signal and status mechanisms. In conjunction with the AC it controls the conceptual image available to each p-user.

Defined PE (2.3.2, 2.3.6)

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

Delivery Control (2.3.7.7, GVT)

The facility in VTS which causes updates to the access controlled parts of the CCA made by the "updating" p-user (the one having access rights to the CCA) to be withheld from the other "observing" p-user until the updating p-user designates explicitly the point in the activity on the CCA at which the new state and content of the CCA is to be made available to the other p-user. This is done with no change of access rights. Acknowledgement of receipt can be solicited from the other p-user. (Delivery is implicitly forced if access right is transferred but acknowledgement receipt is then not available).

Destructive (in the context of effects of services)

See Appendix C.

Dimensions (2.2.3)

The number of dimensions defines the organization of the cells in the CDS.

Directed Graph for Parameters (2.3.6.3)

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

Disestablishment Phase (2.3.2, 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 (2.3.2)

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 activites 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)

The service token used to mediate the issue of requests for a change of enclosure.

Encoding (3.3, 3.4)

Used in the restricted sense in which a character repertoire which has been negotiated is represented by (encoded into) elements in the transfer protocol.

Establishment Phase (2.3.2, 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.

Local Mapping (3.1.1)

Used to refer to the process, completely outside the scope of VTS (and indeed OSI) standards, by which one or both of the p-users perform additional transformations to present the agreed conceptual image of the VTS service on a virtual device as an actual image on a real device. The standard caters only for the imposition of constraints on such local mappings by means of the "realization" parameters agreed for each virtual device.

Move Window (2.2.3.1.4)

The move window determines at any instant the range of co-ordinate values at which information may be written.

Multiple Interaction Negotiation (2.3.6.2)

The type of negotiation used within a negotiation-enclosure to agree or modify the PE definition. During the operation presentation parameters can be negotiated individually or in groups.

Negotiate-token (2.3.6.2)

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

Negotiation (2.3.2, 2.3.6)

The action of redefining the PE.

Negotiation-enclosure (2.3.2, 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)

A set of related parameters currently being negotiated.

Null-enclosure (2.3.2)

The type of enclosure entered after establishment of 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)

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 (2.3.6.3)

The conceptual repository for the address of the currently addressed cell (active location) of the CDS. The pointer holds one, two or three coordinate values depending on the dimensionality negotiated for the CDS. Its initial setting is at the origin (1) on all existing dimensions. It is subsequently moved, explicitly or implicitly, by the p-users subject to access control constraints.

Position Control (2.3.3.1.3)

The means available in the VTS which enable the p-user having access right to the CDS to control the position of the active location in relation to the dimensions of the CDS.

Presentation Environment (2.2)

For Basic Class this is the only complete and self consistent set of values for the parameters of the VT service. It is not necessary for all possible presentation parameters to have values to complete this PE. It can be re-negotiated (i.e. some parameters changed) if the selected operating subset contains negotiation facilities. It ceases to exist when the p-connection is disestablished.

Primary Value (2.2.2)

That portion of the atomic object which determines the character it currently represents. This is probably most easily understood as an index into the agreed character repertoire.

Note that the choice of repertoire, when several have been negotiated, may be one of the character attributes. It is this primary value which is subject to encoding.

Profile (2.3.5.5)

Used in a restricted sense to refer to a pre-established set of parameter values assumed mutually known to the two p-users (and therefore conceptually preset in the CCA). A profile can be referenced during connection establishment or negotiation to short cut the individual negotiation of large numbers of parameters.

Protocol Message (3.1.2)

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

Protocol Message Element (3.4)

In Basic Class version 1 the encoding of P-DATA follows ECMA-48. This in conjunction with standards ECMA-6, ECMA-35 and ECMA-43 defines a transfer syntax with graphic characters, control functions, escape sequences, etc. These are referred to protocol message elements.

Realization [with respect to logical attributes and mechanisms] (2.2.7)
A set of negotiable parameters allow one peer entity to control the way in which logical attributes, mechanisms, layouts, etc., are mapped onto the real devices associated with the logical ones. These facilities are referred to as realization control.

Rendition (2.2.5)

Used in the restricted sense of a style of visual presentation of the characters with which Basic Class is concerned.

Root Parameter (2.3.6)

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 (2.3.1)

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 (2.3.1)

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 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 (2.2.1, 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 (2.3.7)

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 Device (2.1, 2.2.7)

One (or the only) conteptual device on which the p-users visualize a conceptual image being displayed when using the Virtual Terminal Service. The CCA and its components are the essential part of the model. The interpretation of the contents of the CCA as a conceptual image and the one or more virtual devices on which the p-users visualize this conceptual image being displayed are no more than alternative interpretation of the information in the model, in particular they do not add anything, neither data nor flow paths, to the model.

Virtual Terminal Protocol, VTP, (3.1.1)

The transfer syntax standarized for transmission over an Open Systems Interconnection between two users of the Virtual Terminal Service. This is defined in terms of protocol messages, rules for combining these into valid protocol structures and sequences of structures and rules for the packaging of these for exchange between the p-users using the service, by means of a session service facility.

Virtual Terminal Service, VTS, (2.1)

The component of the Open Systems Interconnection model defined by ISO 7498, which is concerned with presenting information normally, though not necessarily, intended for human consumption. Normally the VTS will be used by an application program to present information to a human terminal operator and vice versa. However the associated protocol is defined symmetrically so can be used between two application p-users or between two terminal operator p-users.

Write Access (2.2.1)

The right to perform updates to parts of the Virtual Terminal covered by access control; in Basic Class this right is held exclusively by one user at a time with applicability to all controlled parts of the Virtual Terminal. This right is manifested by possession of the write-access-token. All parts of the CCA not covered by the token are uncontrolled for update; however it may not be meaningful for one or other p-user to actually update any one such item.

B.3 Index of Acronyms

No attempt has been made to cover abbreviations introduced in a localized context for use in tabular presentations.

AC Access Control

AR Acknowledge Receipt*

BNF Backus-Naur Format

CCA Conceptual Communications Area

CDS Conceptual Data Store

CICI Conceptual Interface Command Interpreter

CE Change Enclosure*

CN Connect

CSS Control, Signalling and Status (applied to area or mechanism)

DL Deliver*

DN Disconnect*

DSD Data Structure Definition

FC Free Control*

GPP Generic Presentation Protocol

GPS Generic Presentation Service

GT Give Tokens*

GVT Generic Virtual Terminal

II Indication-Indication (applied to service structure)

NA Negotiation Accept*

NC Normal Control*

ND Normal Data*

NI Negotiation Invite*

NO Negotiation Offer*

NR Negotiation Reject*

PE Presentation Environment

PN Perform Negotiation*

PPDU Presentation Protocol Data Unit

PPM Presentation Protocol Machine

PSAP Presentation Service Access Point

RC Request-Confirmation (applied to service structure)

RL Release*

RI Request-Indication (applied to service structure)

RO Request Only (applied to service structure)

RT Request Tokens*

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

* *In the context of xxQ Request Message of xxR Response Message.*

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.

APPENDIX D

FORMAL DESCRIPTION OF PROTOCOL

D.1 Introduction

This Appendix is an integral part of the standard.

In Section 3 of the standard, the virtual terminal protocol interactions between two VTS presentation entities are described. That description references states, events and actions which in this Appendix are consolidated into a formal description of the protocol.

D.2 Elements Used in the Formal Description

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

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

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

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

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

Table D/1 - VTM States

State code	State Description	State Tables	Subsets
CF..	(Connection Facility States)		
CF01	Unconnected	D/6	A B C D
CF02	CN-pending	D/6	A B C D
CF03	RL-pending	D/6	A B C D
CF04	Null-enclosure	D/6, 7, 8, 10	D
NF..	(Negotiation Facility States)		
NF01	Perform-neg'n pending	D/6, 7, 10	C D
NF02	Negotiation State	D/6, 8	D
NFi1	Parameter i Fixed External	D/6, 12	D
NFi2	Parameter i Not Applicable	D/6, 12	D
NFi3	Parameter i Negotiable/Agreed	D/6, 11, 12	D
NFi4	Parameter i Invited	D/6, 11, 12	D
NFi5	Parameter i Offered	D/6, 11, 12	D
NFi6	Parameter i Counter-offered	D/6, 11, 12	D
NFi7	Parameter i Frozen	D/6, 12	D
TF..	(Transfer Facility States)		
TF01	Normal Transfer	D/6, 7, 8, 9, 10	A B C D
TF03	Acknowledge Pending	D/6, 9, 10	B C D
EF..	(Enclosure Facility States)		
EF01	Change Enclosure Pending	D/6, 8, 10	D
..A	owning service token for CF04, NF02, NFi1, NFi2, NFi3, NFi7, TF01. request initiator for CF02, CF03, NF01, TF03, EF01. negotiation cycle initiator for NFi4, NFi5, NFi6.		
..B	not owning service token for CF04, NF02, NFi1, NFi2, NFi3, NFi7, TF01. request acceptor for CF02, CF03, NF01, TF03, EF01. negotiation cycle acceptor for NFi4, NFi5, NFi6.		

Additional information to Table D/1:

A) The state NF02 is a shorthand representation of the outer product of the parameter states (NFi1 through NFi7) of all applicable parameters (i.e. all parameters that may exist in the PE) at any moment during the negotiation-enclosure.

The detailed value of this parameter immediately after entering the negotiation-enclosure is determined by the agreed draft PE, but consists only of the states NFi1, NFi2 and NFi3.

The detailed value of this parameter allowing exit from the negotiation-enclosure depends on the value of the pp-pe-action parameter. The value "restore" allows exit for any value of NF02;

The value "switch" is only allowed if NF02 reflects a consistent presentation environment.

B) In order to simplify the state diagrams the interpretation of the postfixes A and B depends on the protocol states as indicated in the bottom part of Table D/1.

In general A and B are assigned meanings such that one of the protocol machines goes through the A states while the other goes throug the corresponding B states.

Table D/2 - VTM Events

Event-code	Event Description
XXQ	XX request message
XXR	XX response message
XX-RQ	request primitive associated with XX
XX-IN	indication primitive associated with XX
XX-RP	response primitive associated with XX
XX-CF	confirmation primitive associated with XX
S.C.AB	session connection abort
AB-IN	abort indication

Additional information to Table D/2:

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

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

Table D/3 - List of Protocol Messages

Acronym	Message Name	Type	State Tables	Subsets
CN	CONNECT	2	D/6	A B C D
RL	RELEASE	2	D/6	A B C D
DN	DISCONNECT	1	D/6	A B C D
GT	GIVE TOKEN	1	D/10	A B C D
RT	REQUEST TOKEN	1	D/10	A B C D
PN	PERFORM NEGOTIATION	2	D/7	C D
CE	CHANGE ENCLOSURE	2	D/7	D
NIi	INVITE PARAMETER i	1	D/11, 12	D
NOi	OFFER PARAMETER i	1	D/11, 12	D
NAi	ACCEPT PARAMETER i	1	D/11, 12	D
NRi	REJECT PARAMETER i	1	D/11, 12	D
ND	NORMAL DATA	1	D/9	A B C D
NC	(NORMAL) CONTROL	1	D/9	A B C D
FC	FREE CONTROL	1	D/9	A B C D
DL	DELIVER	1	D/9	A B C D
AR	ACKNOWLEDGE RECEIPT	1	D/9	A B C D

For messages of type 1 only the events XXQ, XX-RQ, XX-IN exist.

For messages of type 2 all events with XX exist.

For messages of type 5 only the event XXQ exists.

Table D/4 - Conditions

Condition	Meaning
+	value of diagnostic severity is "success" or "success with warning".
-	value of diagnostic severity is "failure".
Ai	condition; forcing alternate actions by both users for parameter i during negotiation Ai: most recent action from Neg Cycle initiator °Ai: most recent action from Neg Cycle acceptor.
Bi	condition; indicating if "set" that for parameter i a Negotiation Cycle is started but that the Negotiation Group is not yet established.
Ti	condition; indentifying if "set" that parameter i is member of a Negotiation Group.
Ri	condition; identifying if "set" that parameter i is the root member of a Negotiation Group.

	Note that there is one condition A, B, R and T for each parameter.
C1	condition; "set" if "no complete PE is agreed as result of the CNQ and CNR messages" and "subset is D", "reset" otherwise.
C2	condition; "set" if "a complete PE is agreed as result of the CNQ and CNR messages", "reset" otherwise.
	Note that a PE may become complete due to the default values of parameters; in particular agreement on the parameter pp-protocol-definition is sufficient to have an agreed PE.
ar	condition; "set" if Ack Rec is requested, else "reset"
d	condition; "set" if delivery-control option agreed, else "reset".
ti	condition, one for each CSS area; "set" if CSS area "i" is a trigger mechanism, else "reset".

D.3 Formal Description Conventions

The formal description is in tables D/6 through D/14.

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

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

Each intersection is either empty, in which case it is always Invalid, or it contains:

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

The conditions are those defined in Table D/4.

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

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

If, for an occurrence of a conditional intersection, the conditions are met then the occurrence is Valid, the actions are performed and the new state entered. If the conditions are not met the occurrence is Invalid.

All invalid intersections and invalid occurrences of conditional intersections between states and incoming message events are treated as protocol violations: the action is to disconnect the VTS connection (DNQ message with "protocol violation" diagnostic), the new state is CFO1 (unconnected).

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

In some intersections, whether or not the occurrence of the intersection is itself conditionally or unconditionally valid, there can be actions or state changes which are conditional.

The Formal Definition for a particular subset consists of the intersection in the Formal Definition tables of the states existing in that subset (see Table D/1) and the events existing in that subset (see Tables D/2 and D/3).

The following editorial conventions are used:

: = action with regard to state change
, = connects list of actions
+ = actions combined as appropriate
a&b = logical "and" of conditions a and b
° = logical negation

ST(state) RS(state)	Each presentation entity is considered to have a (one) position in which its current state can be stored (ST), to be reinstalled later (RS).
|ND-IN|	Only the net-effect of the NDQ messages is given.
ST(CCA) ST(CDS) ST(CCS)	The information transferred is stored in the indicated part of the virtual terminal.

D.4 Formal Description Tables

The complete state table for the Virtual Terminal Protocol VTM is too large to be edited on a single page. In addition it contains a majority of empty cells, representing error cases. Therefore, this state table has been segmented into smaller subtables; sub-tables with only empty cells have not been represented.

It should be understood that the validity of events with respect to one particular Presentation Connection is indicated.

Table D/5 - Index of Formal Description Tables

TABLE	SUB-PROTOCOL
D/6	Connection
D/7	Perform Negotiation Facility
D/8	Change Enclosure Facility
D/9	Normal Data, Control Data, and Delivery
D/10	Token Transfer (excluding negotiation)
D/11	Negotiation of Parameter
D/12	Effect of Parameter Negotiation on other Parameters
D/13	Token Transfer in Negotiation Enclosure
D/14	Structuring Rules, etc., for Negotiation Group

Table D/6

	CF01	CF02A	CF02B	CF03A	CF03B	CF04A
CN-RQ	:CF02A CNQ					
CNQ	CN-IN :CF02B					
CN-RP			- :CF01 +&C1:CF04B +&C2:TF01B CNR			
CNR		CN-CF -:CF01 +&C1:CF04A +&C2:TF01A				
RL-RQ						:CF03A RLQ
RLQ						ST(state) :CF03A RLQ
RL-RP					+:CF01 -:RS(st) RLR	
RLR				RL-CF +:CF01 -:RS(st)		
DN-RQ		:CF01 DNQ	:CF01 DNQ	:CF01 DNQ	:CF01 DNQ	:CF01 DNQ
DNQ		DN-IN :CF01	DN-IN :CF01	DN-IN :CF01	DN-IN :CF01	DN-IN :CF01
S.C.AB		AB-IN :CF01	AB-IN :CF01	AB-IN :CF01	AB-IN :CF01	AB-IN :CF01

Table D/6

	CF04B	TF01A	TF01B	ALL OTHER STATES
CN-RQ				
CNQ				
CN-RP				
CNR				
RL-RQ		ST(state) :CF03A RLQ		
RLQ	RL-IN ST(state) :CF03B		d:|ND-IN|+ CD-IN RL-IN ST(state) :CF03B	
RL-RP				
RLR				
DN-RQ	:CF01 DNQ	:CF01 DNQ	:CF01 DNQ	:CF01 DNQ
DNQ	DN-IN :CF01	DN-IN :CF01	DN-IN :CF01	:CF01
S.C.AB	AB-IN :CF01	AB-IN :CF01	AB-IN :CF01	AB-IN :CF01

Table D/7

	CF04A	CF04B	TF01A	TF01B	NF01A	NF01B
PN-RQ	:NF01A ST (state) PNQ		:NF01A ST (state) PNQ			
PNQ		PN-IN ST (state) :NF01B		d:|ND-IN|+ |CD-IN| ST (state) PN-IN :NF01B		
PN-RP						:RS(state) PNR see note
PNR					PN-CF :RS(state) see note	

Depending on diagnostic the PE is either changed or unchanged. If return is to TF01, the changed PE is installed immediately or use of the unchanged PE is resumed in the state from which it was left, i.e. it is not initialized.

Table D/8

	CF04A	CF04B	TF01A	TF01B
CE-RQ	:EF01A ST(state) CEQ		:EF01A ST(state) CEQ	
CEQ		CE-IN ST(state) :EF01B		d:|ND-IN|+|CD-IN| ST(state) CE-IN :EF01B
CE-RP				
CER				

	NF02A	NF02B	EF01A	EF01B
CE-RQ	:EF01A ST(state) CEQ			
CEQ		CE-IN ST(state) :EF01B		
CE-RP				:see below CER
CER			CE-CF : see below	

The state adopted after CE-RP or CER is indicated in the parameter pp-dest-encl-type of the response message. It can only be CF04 or TF01 or NF02:

- if exit and entry conditions are correct that state specified in the pp-dest-encl-type of the CEQ message is adopted,
- if exit conditions are correct but entry conditions are incorrect, the null-enclosure is entered,
- if exit conditions are incorrect the state stored in the "state" variable is entered.

Table D/9

	TF01A	TF01B	TF03A	TF03B
ND-RQ	ST(CDS) :TF01A NDQ			
NDQ		ST(CDS) °d:ND-IN :TF01B		
NC-RQ	ST(CSS) °ti:TF01A NCQ ti:TF01B NCQ+GTQ			
NCQ		ST(CSS) °d:NC-IN :TF01B		
FC-RQ	ST(CSS) :TF01A FCQ	ST(CSS) :TF01B FCQ	ST(CSS) :TF03A FCQ	ST(CSS) :TF03B FCQ
FCQ	ST(CSS) FC-IN :TF01A	ST(CSS) FC-IN :TF01B	ST(CSS) FC-IN :TF03A	ST(CSS) FC-IN :TF03B
DL-RQ	°ar:TF01A ar:TF03A DLQ			
DLQ		d:|ND-IN|+|CD-IN| DL-IN °ar:TF01B ar:TF03B		
AR-RQ				:TF01B ARQ
ARQ			AR-IN :TF01A	

NOTE: The FD tables indicate that without delivery-control ND and NC messages are immediately delivered. This delivery may be deferred provided the sequential delivery is guaranteed.

Table D/9

	NF01A	CF03A	EF01A
ND-RQ			
NDQ			
NC-RQ			
NCQ			
FC-RQ			
FCQ	ST(CSS) FC-IN :NF01A	ST(CSS) FC-IN :CF03A	condn: ST(State)=TF01A ST(CSS) FC-IN :EF01A
DL-RQ			
DLQ			
AR-RQ			
ARQ			

Table D/10

	CF04A	CF04B	EFO1A	NF01A
GT-RQ	:CF04B GTQ	:CF04B		
GTQ		GT-IN :CF04A		
RT-RQ	:CF04A	:CF04B RTQ		
RTQ	RT-IN :CF04A	RT-IN :CF04B	condn:ST(state)= TF01A or CF04A RT-IN :EF01A	RT-IN :NF01A

	TF01A	TF01B	TF03A	TF03B	NFij	CF03A
GT-RQ	:TF01B GTQ	:TF01B			see D/13	
GTQ		d:|CD-IN|+ |ND-IN| GT-IN :TF01A			see D/13	
RT-RQ	:TF01A	:TF01B RTQ				
RTQ	RT-IN :TF01A	RT-IN :TF01B	RT-IN :TF03A			RT-IN :CF03A

Table D/11

	NFi3A	NFi3B	NFi4A	NFi4B
NIi-RQ :	condn: °Ti :NFi4A set Ai, Bi NIiQ			
NIiQ		condn: °Ti NIi-IN set Ai, Bi :NFi4B		
NOi-RQ	condn: °Ti :NFi5A set Ai, Bi NOiQ			condn: (Ai&nt) :NFi5B set °A NOiQ
NOiQ		condn: °Ti NOi-IN set Ai,Bi :NFi5B	condn: (Ai&°nt) NOi-IN set °Ai :NFi5A	
NAi-RQ				
NAiQ				
NRi-RQ				
NRiQ				

Table D/11

	NFi5A	NFi5B	NFi6A	NFi6B
NIi-RQ				
NIiQ				
NOi-RQ	condn:°Ai&nt :NFi6A set Ai NOiQ	condn:Ai&nt :NFi6B set °Ai NOiQ		
NOiQ	condn:Ai&°nt NOi-IN set °Ai :NFi6A	condn:°Ai&°nt NOi-IN set Ai :NFi6B		
NAi-RQ	condn:°Ai&nt :NFi3A NAiQ	condn:Ai&nt :NFi3A NAiQ	condn:°Ai&nt :NFi3A NAiQ	condn:Ai&nt :NFi3A NAiQ
NAiQ	condn:Ai&°nt NAi-IN :NFi3B	condn:°Ai&°nt NAi-IN :NFi3B	condn:Ai&°nt NAi-IN :NFi3B	condn:°Ai&°nt NAi-IN :NFi3B
NRi-RQ	condn:°Ai&nt :NFi7A NRiQ reset Ti	condn:Ai&nt :NFi7A NRiQ Reset Ti	condn:°Ai&nt :NFi7A NRiQ reset Ti	condn:Ai&nt :NFi7A NRiQ reset Ti
NRiQ	condn:Ai&°nt NRi-IN :NFi7B reset Ti	condn:°Ai&°nt NRi-IN :NFi7B reset Ti	condn:Ai&°nt NRi-IN :NFi7B reset Ti	condn:°Ai&°nt NRi-IN :NFi7B reset Ti

Table D/12

	NFijA	NFijB
NIn-RQ	:NFijA	:NFijB
NInQ	:NFijA	:NFijB
NOn-RQ	:NFijA	:NFijB
NOnQ	:NFijA	:NFijB
NAn-RQ	:NFijA	:NFijB
NAnQ	:NFijA	:NFijB
NRn-RQ	:NFijA	:NFijB
NRnQ	:NFijA	:NFijB

- for j = 1, 2, 3, 4, 5, 6, 7
- for all parameters n except n = i

Table D/13

	NFi1A	NFi1B	NFi2A	NFi2B	NFi3A	NFi3B
GT-RQ	:NFi1B		if child of new NG :NFi7B else:NFi2B		if parent of new NG :NFi7B if child of new NG :NFi7B if(Ti&state of any param on path to root≠NFi3) then (reset Ti, :NFi7B) else (validate value,:NFi3B) otherwise:NFi3B. In all cases, reset Ti, Ri.	
GTQ		:NFi1A		if child of new NG :NFi7A else:NFi2A		if parent of new NG :NFi7A if child of new NG :NFi7A if(Ti&state of any param on path to root ≠NFi3) then (reset Ti, :NFi3A) else (validate value,:NFi3A) otherwise:NFi3A. In all cases, reset Ti, Ri.

Refer to additional information following the last part of the Table.

Table D/13

	NFi4A	NFi4B	NFi5A	NFi5B	NFi6A	NFi6B
GT-RQ	if state of any param on path to root ≠ NFi4 :NFi7B | :NFi7A reset Ti else :NFi4B | :NFi4B		if state of any param on path to root ≠ NFi5 :NFi7B | :NFi7A reset Ti else :NFi5B | :NFi5B		if state of any param on path to root ≠ NFi6 :NFi7B | :NFi7A reset Ti else :NFi6B | :NFi6B	
GTQ	if state of any param on path to root ≠ NFi4 :NFi7B | :NFi7A reset Ti else :NFi4A | :NFi4A		if state of any param on path to root ≠ NFi5 :NFi7B | :NFi7A reset Ti else :NFi5A | :NFi5A		if state of any param on path to root ≠ NFi6 :NFi7B | :NFi7A reset Ti else :NFi6A | :NFi6A	

Refer to additional information following the last part of the table.

Table D/13

	NFi7A	NFi7B	NF02A	NF02B
GT-RQ	for, parents of agreed root, without other negotiated descendents; :NFi3B for, descendents of accepted root; - assign default, - :NFi2B or :NFi3B for, descendents of rejected root and root; - original value, - :NFi2B or :NFi3B, - reset R. In all other cases: :NFi7B		for all i; if "Bi" & "parent≠Bi" then set Ri. condn: for each (new) Ri "all params between "Ri" and "root of param tree" must be°"Ti" if Bi then set Ti, reset Bi. ---------- GTQ	
GTQ		for, parents of agreed root, without other negotiated descendents; :NFi3A for, descendents of accepted root; - assign default, - :NFi2A or :NFi3A for, descendents of rejected root and root; - original value, - :NFi2A or :NFi3A - reset Ri in all other cases; :NFi7A		for all i; if "Bi" & "parent≠Bi" then set Ri. condn: for each (new) Ri "all params between "Ri" and "root of param tree" must be °Ti" if Bi then set Ti, reset Bi. ------ GT-IN

Additional Information to Table D/13

This table represents only the states NF02A and NF02B. The states NF02 are the outer product of all the states of the individual parameters. Each column of the table represents one of the states that can be assumed by a parameter: in addition there are two global columns (one for each presentation entity).

If one of the events indicated occurs then the following actions and state transitions are made:

first: the actions/transitions described in the appropriate global column, above the horizontal dashed line,

second: top to bottom, for each parameter: the actions/transitions as described for the actual state of the parameter,
(note that during the complete step the presentation entity is aware of:

- new roots,
- accepted roots,
- rejected roots,
- other root parameters,

so that for all parameters the hierarchical relation with respect to the roots is known, even if "Ri" conditions have already been reset).

third: generation of the GTQ message or GT-IN event as applicable.

Table D/14 - Structure Conditions during Negotiation

1. Negotiation in the Directed Graph structure occurs for individual parameters and/or groups of parameters. The set of negotiated parameters consists of one or more Negotiation Groups (NG). A parameter can at most be a member of one NG.

2. All parameters of an NG are linked with each other via the parent/child links of the Directed Graph and form a hierarchical ordered set with one and only one root parameter. All parameters of each link between two members of an NG belong to that NG.

3. A Negotiation Group is established at the first token passing (of the negotiate-token) following the start of a Negotiation Cycle for a parameter; amongst others the root and the members of the NG are determined. An NG ceases to exist at the first token passing following the acceptance or rejection of the negotiation for its root. When an NG ceases to exist the agreed parameter values are checked for consistency.

4. No parameters can be added to an NG after it has been established.

5. All elements of an NG should follow the same Negotiation Cycle, i.e. should have the same state as the root element when the token is passed. This rule has the following exceptions:

   - parameters in the NG which, during token passing, do not have the same state as the root, and all descendents of such parameters, become "frozen" on the first subsequent token passing and are no longer regarded as part of the NG;
   - parameters which are accepted become "agreed/negotiable"; parameters which are rejected become "frozen". These states may coexist until the first subsequent passing of the token, at which moment the NG will cease to exist.

6. No Negotiation Cycle may be started for the parents of an NG. Parents will be "frozen" when the NG is established, unless they are external. Parents are brought back to their original state when the NG ceases to exist, unless any other of their descendents is a member of (another) NG.

7. No Negotiation Cycle may be started for the children of an NG. They will be set to "frozen" when the Negotiation Cycle is established. When the NG ceases to exist they will be:

   - reset to their original state if the root has been rejected,
   - set to default according to the table of parameters in sub-clause 2.3.6.4 if the root is accepted.

APPENDIX E

MAPPING OF REAL TERMINALS AND PROFILE DESCRIPTIONS

E.1 Introduction

This Appendix gives some examples of real devices and the equivalent Virtual Terminal type. In addition there is descriptive material directly related to the profiles defined in sub-clause 2.3.6.5.2.

This Appendix is not part of the Standard and not subject to any conformance requirements.

During the establishment and/or negotiation phases the p-users must agree on the characteristics (the presentation environments) of the Virtual Terminal. These characteristics have been designed to enable the features to be closely defined including aspects of mapping onto real devices (e.g. move windows, minimum available dimension, actual rendition mapping). However, no conformance requirements are defined for the actual realization of these features so that this may differ from one implementation to another. It is open to users of the service to use an independent management function prior to connection to ensure that the real resources and characteristics are actually available.

E.2 Notes on the Mapping of Certain CCA Features

Cell Primary Values

The character symbol set available on the real device need not necessarily exactly conform to that agreed for use during the VTP transfer environment. It may for example be satisfactory for lower case characters to be mapped onto upper case or for certain symbols that are not available to be displayed as a default symbol.

Emphasis Attributes

The level of emphasis attribute, when used without the emphasis realization, provides a flexible mechanism which can be mapped onto actual realization under control of the local implementation and/or human user.

In some instances the negotiated number of emphasis levels may be less than those actually available on the real device so that the operator could choose the most suitable mapping (e.g. whether to use flashing, or reverse video or underline).

Conversely, where the number of emphasis realization is less than the number negotiated, the human user may be able to manage with several mapped onto one real mechanism.

The use of the level of emphasis attribute without the emphasis realization is recommended wherever possible since it allows the maximum freedom for local mapping functions and thus the greatest chance of obtaining satisfaction from a potential partner.

Where a range of colours is available on the real device but is not agreed as a parameter, the colour range may be used as the emphasis realization. Conversely, where colour is an agreed parameter but is not available on the real device, it may be acceptable to map the colour range onto grey scales (see notes to table in sub-clause 2.3.6.4).

CSS Areas and Mechanisms

The CSS descriptors provide a general method of defining data structures which are used for the transfer of control, signalling or status, or similar information, between the p-users.

Apart from the broad classification, the mapping of the negotiated feature onto a particular mechanism on the real device is not defined and a pre-agreed understanding of their use would normally be necessary.

Typical of such understanding would be their use as an annunciation panel of lights, or for a set of function keys. In both cases the feature could be agreed to be independent of the write access token and the program function keys could cause a trigger.

E.3 Terminal Types and Profile Descriptions

Line Mode Relative Addressing Terminals Profile P1 (r1)

P1 has only one parameter (r1) which caters for varying line length. The two-dimensional CDS with X bounded (p21.1=r1) and Y unbounded allows only relative forward addressing in X and forward relative movement of 1 in Y (the "window" is always a single line). Movement to the start of the next line (i.e. y:=y+1, x:=1) is permitted although no other absolute X addressing is available. There is no delivery-control, no erasure, only encoding, with the IRV character repertoire.

A single output-only virtual device is associated with the CCA with "unspecified" font and only the automatically supplied CSS area giving on/off control, interrupt, status alert and alarm capability. Subset VTB-A is the operating subset selected by this profile.

Page Mode Relative Addressing Terminals Profile P2 (r1, r2, r3)

P2 has three parameters (r1, r2, r3) catering for varying screen width and depth and allowing a hard copy auxiliary device (the second virtual device appears only if r3 = "yes"). The 3-dimen-

sional CDS has X bounded (p21.1=r1) and Y bounded (p22.1=r2) Z being unbounded. Only forward relative addressing is allowed and movement in Z is restricted to 1 (i.e. the "window" is always a single page). A movement to the start of the next page is permitted (z:=z+1, y:=1, x:=1) although no other absolute addressing of X and Y is available. There is no delivery-control, no erasure, and only embedded encoding. The IRV character repertoire is provided.

A device (screen) is always associated with the CCA and if r3 = 1 a virtual device (hard copy printer) is also associated. The virtual device(s) provide only "unspecified" font with two levels of intensity, "normal" and "high" (the precise meaning of "high" being otherwise unspecified - it may be mapped to "bold", "bright", "underline", "inverse video", "blink" or whatever else is available). No layout distortion is allowed on either device (p78.1 = r1, p78.2 = r2). Each device has only its default CSS area giving on/off control, interrupt, status alert and alarm capability.

If any of these pre-defined capabilities are unsatisfacory the connection request must override (explicitly) the implied VTB subset (="A") in order to be able to negotiate any desired changes or additions to the profile as specified, e.g. if more sophisticated device control or status information is required it must be negotiated.

Full Screen Page Mode Monochrome Terminal Profile P3 (r1, r2)

P3 has three parameters (r1, r2, r3) allowing for varying screen widths and depth and for the addition of a hard copy auxiliary device. It provides a single page (2-dimensional) CDS with X bounded (p21.1=r1) and Y bounded (p22.2=r2) but with full forward relative and absolute addressing. Three emphasis levels are provided. There is no delivery-control, no erasure and only embedded encoding. The IRV character repertoire is provided.

A virtual device (screen) is associated with the CCA and if r3 = 1 a second virtual device (auxiliary hard copy printer) is added. The device(s) provide one unspecified font with three levels of intensity, "invisible", "normal" and "high" (the precise realization being otherwise unspecified, so that for example "invisible" may be blank or obliterated). No layout distortion is allowed (p78.1=r1, p78.2=r2) and only default CSS on/off control, interrupt, status alert and alarm capabilities are automatically provided within the profile. Subset VTB-A is selected as operating subset.

As in the earlier profiles, if deviations are to be negotiated the connection request must explicitly override the implied subset VTB-A.

Full Screen Page Mode Colour Terminal Profile P4 (r1, r2, r3)

P4 has three parameters (r1, r2, r3) allowing varying width, depth and page capacity. It provides a multi-page 3-dimensional CDS with X bounded (p21.1 = r1), Y bounded (p22.1 = r2) and Z bounded (p23.1 = r3). Complete freedom of addressing is allowed in all dimensions (any window smaller than the whole CDS must be negotiated). The IRV character repertoire is provided, with three levels of emphasis and seven colour foreground capabilities (the 8 colour values include black). Delivery-control and erasure capability are included. Subset VTB-C is implied offering single-interaction negotiation from the profile without the need to override at establishment time (if full negotiation, i.e. including multiple-interaction type, is required, subset VTB-D must be selected at establishment time, or later subject to subset not being "fixed" at establishment time). Encoding is specified. One virtual device (a full colour screen) is associated with the CCA (availability of hard copy colour printers is not yet sufficiently widespread to warrant inclusion in standard profile).

The device provides one unspecified font "as available" with intensities "invisible", "normal" and "high" and with a standard list of 7 colours plus black. No layout distortion is allowed (p78.1 = r1, p78.2 = r2) and only the minimum default CSS area is automatically provided.

E.4 Further Examples of Real Device Mapping

A. Stream Type Device, e.g. Paper Tape

An appropriate presentation environment for a stream type real device could be a CDS of one unbounded (X) dimension, with backwards move and home on X (probably) not possible.

An alternative where there is some form of terminator on the real device or the medium defining some message structure could be to use an unbounded X dimension with each message occupying one X-array of approriate size for it, and an unbounded Y representing the move from one message to the next.

B. Line Type Device, e.g. Character Printer, Card Reader, VDUs Used in Certain Modes

These devices have a direct equivalent to a bounded X dimension but can have an unbounded Y dimension (non paging printer, card reader) or a bounded Y (VDUs used in "page" mode).

Thus an approriate presentation environment could have a CDS with a bounded X dimension and an unbounded Y dimension, possibly without backwards move on Y, so that either p-user can only address the Y dimension in terms of current or next or future "lines". Thus each card from a card

reader could form one bounded X-array with the pack of cards forming the unbounded Y; normally it would be impossible to read backwards or only to small extent. (A further extension could consider a new pack of cards to be a move in an unbounded Z dimension).

An alternative PE is a CDS unbounded in X and Y dimensions. This could be appropriate for an implementation which can handle complete paragraphs of text as variable length X-arrays, mapping a paragraph onto some appropriate number of lines on the real device, with "new paragraph" being a move on the Y dimension to the start of a new X-array.

C. Page Type Devices, e.g. VDU, Page Printer

An example of a 3-dimensional device equivalent to B above has an unbounded X-array capable of handling a paragraph of any length (but probably with a limit of movement backwards), an unbounded Y-array capable of handling any number of paragraphs, and a Z dimension corresponding to chapters or documents.