International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

T.126

(08/2007)

SERIES T: TERMINALS FOR TELEMATIC SERVICES

Data protocols for multimedia conferencing

Multipoint still image and annotation protocol

ITU-T Recommendation T.126

ITU-T T-SERIES RECOMMENDATIONS

TERMINALS FOR TELEMATIC SERVICES

Facsimile – Framework	T.0–T.19
Still-image compression – Test charts	T.20–T.29
Facsimile – Group 3 protocols	T.30–T.39
Colour representation	T.40–T.49
Character coding	T.50–T.59
Facsimile – Group 4 protocols	T.60–T.69
Telematic services – Framework	T.70–T.79
Still-image compression – JPEG-1, Bi-level and JBIG	T.80–T.89
Telematic services – ISDN Terminals and protocols	T.90–T.99
Videotext – Framework	T.100–T.109
Data protocols for multimedia conferencing	**T.120–T.149**
Telewriting	T.150–T.159
Multimedia and hypermedia framework	T.170–T.189
Cooperative document handling	T.190–T.199
Telematic services – Interworking	T.300–T.399
Open document architecture	T.400–T.429
Document transfer and manipulation	T.430–T.449
Document application profile	T.500–T.509
Communication application profile	T.510–T.559
Telematic services – Equipment characteristics	T.560–T.649
Still-image compression – JPEG 2000	T.800–T.849
Still-image compression – JPEG-1 extensions	T.850–T.899

For further details, please refer to the list of ITU-T Recommendations.

ITU-T Recommendation T.126

Multipoint still image and annotation protocol

Summary

ITU-T Recommendation T.126 defines a protocol supporting the management of common multi-layer visual spaces and the multipoint exchange of graphical information directed to these spaces including images (hard and soft copy), pointers, and filled and unfilled parametric drawing elements (points, lines, polygons and ellipses). Support for rendering out-of-band video streams within T.126 workspaces is also included. In addition, keyboard and pointing device exchanges are specified to support basic user interaction. Protocol elements for creating and referencing archived visual spaces are defined to allow pre-stored or pre-distributed graphical materials to be referenced. This protocol uses services provided by ITU-T Recommendations T.122 (MCS) and T.124 (GCC) and complies with the guidelines specified in ITU-T Recommendation T.121 (GAT).

This revised version of ITU-T Recommendation T.126 introduces a number of clarifications to the previous version.

Source

ITU-T Recommendation T.126 was approved on 29 August 2007 by ITU-T Study Group 16 (2005-2008) under the ITU-T Recommendation A.8 procedure.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure e.g. interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

CONTENTS

		Page
1	Scope	1
2	Normative references	2
3	Definitions	4
4	Abbreviations and acronyms	5
5	Overview	5
	5.1 SI application enrolment	5
	5.2 Capabilities and profiles	5
	5.3 Workspaces	5
	5.4 Hard copy devices	7
	5.5 Bitmaps	7
	5.6 Pointers	7
	5.7 Video windows	7
	5.8 Text	7
	5.9 Drawn graphical elements	8
	5.10 Remote events	8
	5.11 Archives	8
	5.12 Conducted mode behaviour	9
6	Use of MCS	9
	6.1 Use of MCS service primitives	9
	6.2 Use of MCS tokens and channels	10
	6.3 Use of MCS data services	11
7	Use of GCC	14
	7.1 Use of GCC services	14
	7.2 GCC unique handles	14
8	Protocol specification	14
	8.1 Session initialization and management	14
	8.2 Interpretation of optional parameters	15
	8.3 SI capabilities	15
	8.4 Workspaces	24
	8.5 Bitmaps	47
	8.6 Pointers	72
	8.7 Video windows	73
	8.8 Text	77
	8.9 Drawn graphical elements	77
	8.10 Remote events	88
	8.11 Archives	93
	8.12 Conducted mode operation	96
9	SIPDU definitions	101

Page

Annex A – SI profiles 133

Annex B – Object identifier assignments 134

Appendix I – Deriving intermediate palettes for bitplane progressive transmission of palettized images 135

ITU-T Recommendation T.126

Multipoint still image and annotation protocol

1 Scope

This Recommendation defines a protocol supporting the management of common multi-layer visual spaces and the multipoint exchange of graphical information directed to these spaces including images (hard and soft copy), pointers, and filled and unfilled parametric drawing elements (points, lines, polygons and ellipses). Support for rendering out-of-band video streams within T.126 workspaces is also included. In addition, keyboard and pointing device exchanges are specified to support basic user interaction. Protocol elements for creating and referencing archived visual spaces are defined to allow pre-stored or pre-distributed graphical materials to be referenced. This protocol uses services provided by ITU-T Recommendations T.122 (MCS) and T.124 (GCC) and complies with the guidelines specified in ITU-T Recommendation T.121 (GAT).

The details of communication with the input and output devices and the user interfaces on the host terminal are considered out of the scope of this Recommendation and are left to the discretion of the implementer. Therefore, this Recommendation makes no assumption that these I/O devices are of any specific architecture.

Figure 1-1 presents an overview of the scope of this Recommendation and its relationship to the other elements of the T.120 framework within a single node.

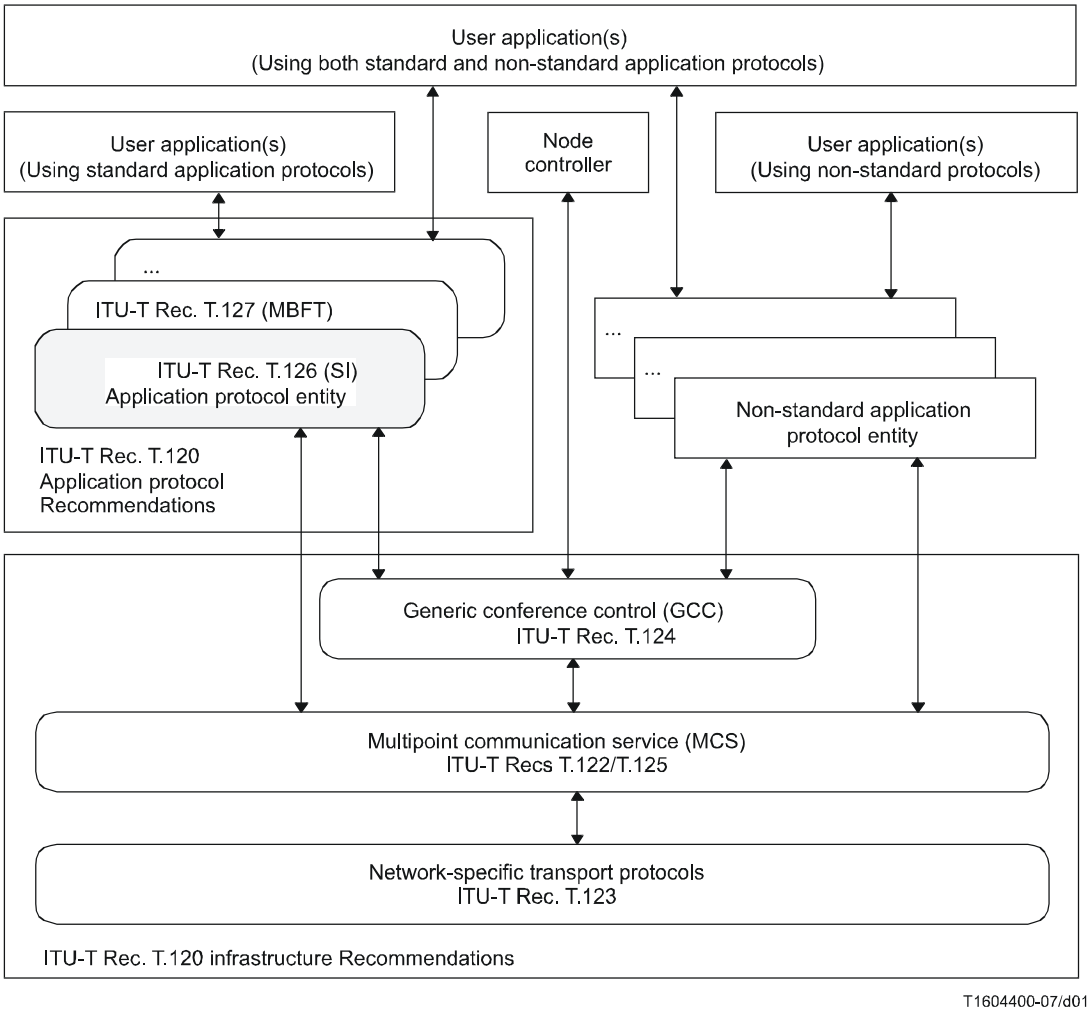

Figure 1-1 – Scope of T.126

2 Normative references

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T F.702] ITU-T Recommendation F.702 (1996), *Multimedia conference services*.

[ITU-T H.221] ITU-T Recommendation H.221 (2004), *Frame structure for a 64 to 1920 kbit/s channel in audiovisual teleservices*.

[ITU-T H.245] ITU-T Recommendation H.245 (2006), *Control protocol for multimedia communication*.

[ITU-T T.4] ITU-T Recommendation T.4 (2003), *Standardization of Group 3 facsimile terminals for document transmission*.

[ITU-T T.6] ITU-T Recommendation T.6 (1988), *Facsimile coding schemes and coding control functions for Group 4 facsimile apparatus*.

[ITU-T T.35] ITU-T Recommendation T.35 (2000), *Procedure for the allocation of ITU-T defined codes for non-standard facilities*.

[ITU-T T.42] ITU-T Recommendation T.42 (2003), *Continuous-tone colour representation method for facsimile*.

[ITU-T T.50] ITU-T Recommendation T.50 (1992), *International Reference Alphabet (IRA) (Formerly International Alphabet No. 5 or IA5) – Information technology – 7-bit coded character set for information interchange*.

[ITU-T T.81] ITU-T Recommendation T.81 (1992) | ISO/IEC 10918-1:1994, *Information technology – Digital compression and coding of continuous-tone still images – Requirements and guidelines*.

[ITU-T T.82] ITU-T Recommendation T.82 (1993) | ISO/IEC 11544:1993, *Information technology – Coded representation of picture and audio information – Progressive bi-level image compression*.

[ITU-T T.120] ITU-T Recommendation T.120 (2007), *Data protocols for multimedia conferencing*.

[ITU-T T.121] ITU-T Recommendation T.121 (1996), *Generic application template*.

[ITU-T T.122] ITU-T Recommendation T.122 (1998), *Multipoint communication service – Service definition*.

[ITU-T T.123] ITU-T Recommendation T.123 (2007), *Network-specific data protocol stacks for multimedia conferencing*.

[ITU-T T.124] ITU-T Recommendation T.124 (2007), *Generic conference control*.

[ITU-T T.125] ITU-T Recommendation T.125 (1998), *Multipoint communication service protocol specification*.

[ITU-T X.680] ITU-T Recommendation X.680 (2002) | ISO/IEC 8824-1:2002, *Information technology – Abstract Syntax Notation One (ASN.1) – Specification of basic notation*.

[ITU-T X.690] ITU-T Recommendation X.690 (2002) | ISO/IEC 8825-1:2002, *Information technology – ASN.1 encoding rules – Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER), and Distinguished Encoding Rules (DER)*.

[ITU-T X.691] ITU-T Recommendation X.691 (2002) | ISO/IEC 8825-2:2002, *Information technology – ASN.1 encoding rules – Specification of Packed Encoding Rules (PER)*.

[ITU-R BT 601-6] ITU-R Recommendation BT 601-6 (2007), *Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios*.

[ITU-R BT 709-5] ITU-R Recommendation BT 709-5 (2002), *Basic values for the HDTV standards for production and programme exchange*.

[CIE 15.2] CIE 15.2 (1986), *Colorimetry, 2nd Edition*.

[ISO/IEC 10646] ISO/IEC 10646:2003, *Information technology – Universal Multiple-Octet Coded Character Set (USC)*.

[ISO/IEC 13818-6] ISO/IEC 13818-6:1998, *Information technology – Generic coding of moving pictures and associated audio information – Part 6: Extensions for DSM-CC*.

3 Definitions

This Recommendation defines the following terms:

3.1 4:4:4: A notation that defines the relative horizontal resolution of a three-colour component raster to be equal.

3.2 4:2:2: A notation that defines the relative horizontal resolution of a three-colour component raster to have twice the horizontal resolution on the first channel than the other two.

3.3 4:2:0: A three-colour component raster having twice the horizontal resolution as well as twice the vertical resolution on the first channel.

3.4 annotation: Real-time drawings (freehand drawing, lines, rectangles, ellipses, etc.) and bitmaps (used for text and unsupported graphical elements, for example) shared between peer SICEs.

3.5 bitmap: A rectangular area described by a two-dimensional array of pixels. These pixels can be coded using a variety of encoding methods.

3.6 control points: A set of points defined in terms of the workspace coordinate system that define a drawing shape parametrically.

3.7 drawing: A type of annotation consisting of instructions for creating points, polylines, rectangles, ellipses or non-standard drawing elements.

3.8 handle: A session-wide unique number used to identify an addressable item.

3.9 image: Photographic or document-oriented information which is transmitted in the form of an image bitmap.

3.10 non-standard capability: The capability is outside the scope of this Recommendation but it has been determined through negotiation that it is recognized among all session participants.

3.11 palette: A finite set of colours defined by at least three linearly-independent colour primaries.

3.12 palettized: A term used to describe visible objects (annotation bitmaps, drawing elements) comprised of palettized pixels. The colour of a palettized pixel is specified by the colour value at the location in a colour look-up table referenced by the pixel value.

3.13 plane: A virtual area defined to have the same pixel dimensions as the workspace with which it is associated. A plane provides a canvas for the use of annotation tools such as drawing, erasing and text, as well as for bitmaps.

3.14 pointer plane: A virtual area defined to have the same dimensions as the workspace with which it is associated. This virtual area in front of all other planes houses all the pointers referencing a given workspace.

3.15 pointer: A bitmap that is moveable over the workspace that is used by its creator as an indicator of position.

3.16 standard capability: The capability is defined within the scope of this Recommendation but is not required for all SICE implementation. Note that all standard capabilities must be negotiated before use.

3.17 still image conference entity: An application protocol entity that interacts with a user application above and with the local multipoint communication service (MCS) and local GCC provider below. Data are exchanged between peer SICEs using still image protocol data units (SIPDUs).

3.18 unicode: Multilingual text string format as defined in [ISO/IEC 10646].

3.19 workspace: A workspace is an area comprising N independent but coincident planes of the same pixel dimensions. The assembly of N planes forms the complete display. At a given workspace coordinate, data in any plane hides data present in underlying planes in the stack unless the pixel value at that plane is transparent. If there are no data at a specific pixel location in the middle or front planes, the location is said to be transparent and does not hide the data from underlying planes.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

GCC Generic Conference Control

GCCSAP Generic Conference Control Service Access Point

MCS Multipoint Communication Service

MCSAP Multipoint Communication Service Access Point

MCU Multipoint Control Unit

PDU Protocol Data Unit

SICE Still Image Conferencing Entity

SIPDU Still Image Protocol Data Unit

5 Overview

5.1 SI application enrolment

An SI application enrols via the application enrolment mechanism specified in [ITU-T T.121].

5.2 Capabilities and profiles

The transactions defined by the SI protocol and the ranges on many of the associated parameters are governed by the capabilities set that is in effect at the time of the exchange. Capability profiles exist for terminals wishing to operate as a whiteboard only, operate as a soft copy image exchange terminal, operate as a soft copy image exchange terminal that can annotate and whiteboard, or operate as a hard copy image exchange device. For forward compatibility reasons, these profiles are simply lists of capabilities that must be advertized; therefore, it is possible for a terminal to be capable of one or more of the above functions.

See Annex A for a detailed description of the standard SI profiles. The capabilities exchange mechanism is the method by which additional capabilities that are not assumed by the specific application class are negotiated. The vehicle for this negotiation is the GCC application enrolment facility which has a well-defined protocol for this purpose.

5.3 Workspaces

The workspace data structure and associated operators provides a self-contained, platform-independent method for describing, manipulating and maintaining related annotation, pointer and bitmap data. A workspace is composed of N depth-ordered planes. Higher numbered planes (and all their associated contents) are assumed to be in front of their lower numbered counterparts within the same workspace. Each plane's contents may include images and/or annotations depending on how the plane is tagged at creation time. Depending on negotiated capabilities, a workspace may also include a virtual pointer plane which resides above all other planes. See Figure 5-1.

A created workspace in a session has the same pixel dimensions at all sites within the session. Placement of annotations and images, each in their respective planes, is indicated in workspace coordinates. All workspace coordinates are specified in units of whole pixels. The pixel aspect ratio is 1:1 but there is no absolute size associated with a pixel. Coordinates referenced in this Recommendation are shown as (X,Y) where the workspace origin (0,0) is defined to be at the upper left hand corner of the workspace. The lower right hand corner of the workspace has the coordinate value (workspace horizontal dimension – 1, workspace vertical dimension – 1). Even though workspace pixels are defined to have a square aspect ratio, bitmaps may be exchanged whose native resolutions are non-square. The allowable aspect ratios are dictated by the enforcing profile and capability set.

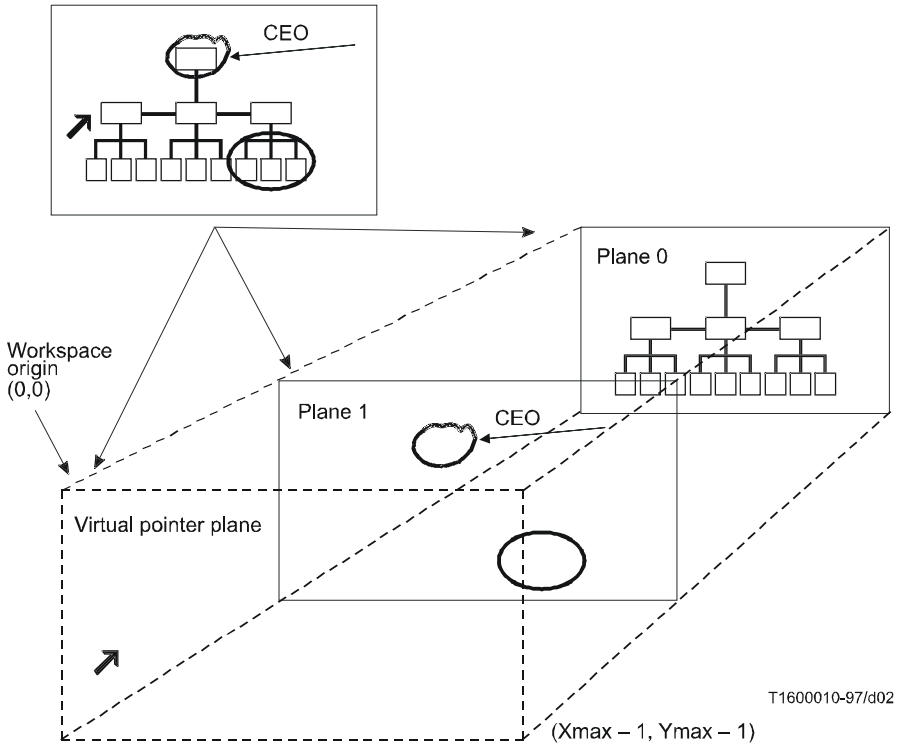

Figure 5-1 – Workspace

To make a workspace visible, one or more workspace views may be defined for a workspace. Each view corresponds to a rectangular region within the workspace to be viewed (which, in the simplest case, is the entire workspace). Within an SI session, only one view among all workspaces is designated as the focus view. This view should be made visible at all nodes.

An SICE implementation may not have a display subsystem that can accommodate the full dimensions of the negotiated maximum size workspace. In this case, the SICE may use any suitable method such as scroll bars or scaling to accommodate a workspace view of a size larger than the local display resolution. Similarly, a workspace view smaller than the local display resolution may, if desired, be scaled appropriately to fill the display. However, regardless of how an SICE chooses to display a workspace view, it must locate its annotations, pointers and bitmaps based on the coordinate system defined for the workspace.

A session can support multiple workspaces given sufficient storage in each SICE. Upon indication by the GCC of new arrivers in the session, all workspaces are deleted upon the creation of the first new one. All SICEs shall create a new workspace whenever they detect that there is a transition from one logical boundary to another. The logical boundary condition is locally determined. An example of one might be the progress from one slide or view graph to the next in a presentation

application using this Recommendation. This Recommendation also defines procedures for refreshing late arrivers with workspace data present in the session before they entered.

5.4 Hard copy devices

An SICE can advertize an optional capability that expresses the ability to receive bitmap exchanges bound for hard copy devices directly. If at least two nodes in the session have this capability, they can exchange these types of bitmaps, which are restricted by a separate capability set from those of the soft copy functions that adapts to facsimile-like devices.

5.5 Bitmaps

The SI protocol supports the exchange of bitmaps. The supported standard formats include:

- uncompressed;
- T.4 (G3);
- T.6 (G4);
- T.81 (JPEG);
- T.82 (JBIG).

Other formats can be negotiated. The governing profile may only allow a subset of the above list in a session. The SI protocol supports bitmap scaling and cropping on a capability negotiated basis. These functions are useful, for example, when attempting to transmit a precompressed image where decompression and recompression would have to be performed if only a subregion of the bitmap is of interest or the bitmap is at an inappropriate resolution.

All submitted bitmaps can be optionally edited and deleted using SI-defined exchanges. These operations are only allowed to workspace planes that will support them and may be disallowed by capabilities negotiation.

5.6 Pointers

Pointers are supported using the bitmap exchange functions. Bitmaps specified as pointers do not belong to any specific workspace plane but instead are managed with respect to the workspace in a virtual plane on top of all other data bearing planes. Pointers are privately held by their creator and are removed if the creator leaves the session in any fashion. Only the creator of a pointer may move it or delete it.

5.7 Video windows

The SI protocol defines exchanges to support the allocation, deletion and management of video window objects that can be placed in SI workspace planes. There is no video information carried via the SI protocol. A referencing mechanism is defined to allow video streams loosely associated with the SI session to be assigned to a video window object. This facility enables conference video streams to be more integrated with the display of graphical information. Features such as interactive video annotation and pointing are examples of functions that are enabled by this service.

5.8 Text

Direct support of text primitives is left for future study. Text can currently be exchanged using the SI protocol by rendering the text locally into a bitmap and then transmitting that bitmap to the session. Alternatively, non-standard text primitives can be negotiated and used within the session. Text bitmaps can use the transparency features of the protocol to support the rendering of only the text information that is drawn into a bitmap.

5.9 Drawn graphical elements

5.9.1 Drawing and erasing basic shapes

The SI protocol defines exchanges for basic drawing shapes. These include:

- open and closed polylines (free hand drawing);
- points;
- rectangles;
- ellipses.

Attributes of the SI basic drawing shapes include:

- line colour;
- fill colour;
- line thickness;
- pen nib shape;
- line style.

Erasing is supported in two ways depending on whether the target workspace plane is of the permanent or editable type. If permanent, erasing is accomplished by drawing over the desired areas with the outline and fill colours set to transparent. Otherwise, erasing is supported by deleting the desired object using the SI drawing delete exchange.

All submitted drawings can be optionally edited and erased using SI-defined exchanges. These operations are only allowable in workspace planes that will support them and may be disallowed from the entire session by capabilities negotiation.

5.9.2 Drawing and erasing custom shapes

Custom shapes that are not supported by the SI protocol or that are disallowed by the specific profile being used or capabilities set negotiated can be supported within the session by rendering them locally into a bitmap which is then submitted to the session using the SI bitmap exchange mechanism. Note that bitmaps can be filled with a transparent colour in all pixel locations that are not affected by the custom shape being rendered.

Custom shapes (and their associated custom attributes) can be supported within a session using non-standard shape and attribute fields within the SI drawing PDUs if they are successfully negotiated.

5.10 Remote events

The SI protocol supports the exchange of keyboard and pointing device events as well as requests for remote workspace printing. These facilities can be optionally supported by terminals if they wish to support basic user interactions with the graphical information presented in SI workspaces (button clicking, basic gesturing, remote printing initiation, etc.). The use of these facilities is not negotiated. These exchanges should be ignored by any terminal that does not support them.

5.11 Archives

This Recommendation supports an archiving function that allows the remote retrieval of information from pre-distributed databases. It also specifies a protocol that can be optionally used to create those databases and add to them remotely.

5.12 Conducted mode behaviour

When a session is in conducted mode, the SICE at the conductor node may grant a set of privileges to perform various actions to one or more nodes in the session. Without such privileges or global conducted-mode permission from the conducting node, an SICE is restricted from performing these actions unless the SICE is the designated refresher for the session.

6 Use of MCS

6.1 Use of MCS service primitives

An SICE uses the following MCS service primitives to attach and detach from a domain, join and leave the SI channel, send and receive SIPDUs, and manage token operations. Table 6-1 describes each of the primitives used by an SICE.

Table 6-1 – MCS primitives needed by an SICE

MCS primitive	Description
MCS-ATTACH-USER	Creates an MCS attachment through an MCS SAP to a domain hosted by the MCS provider. A result is confirmed to the requester. If the request is accepted, a user ID is assigned.
MCS-DETACH-USER	Deletes an MCS attachment that was created previously by invocation of MCS-ATTACH-USER. This primitive may be requested by a user or initiated by a provider. It delivers an indication at every other MCS attachment to the same domain. If provider-initiated, an indication is also delivered at the deleted attachment.
MCS-CHANNEL-JOIN	Used by an application client to join an appropriate channel whose use is defined by the application. This is a prerequisite for receiving data sent to the channel.
MCS-CHANNEL-LEAVE	Used by an application client to leave a previously joined channel and thus stop receiving data sent to that channel. The primitive may be user-initiated (request only) or provider-initiated (indication to affected user only).
MCS-CHANNEL-CONVENE	Used to allocate a new private channel with the requesting user as manager.
MCS-CHANNEL-ADMIT	This primitive enlarges the authorized user group of a private channel at the request of its manager. An indication is delivered to the MCS user added. That user may thereafter send data on the channel or join it as a receiver.
MCS-SEND-DATA	Used to transmit data to other members of a domain. If the sender is a member of the destination channel, it will not receive its own data indications. However, it will receive data indications from other sources addressed to that channel.
MCS-UNIFORM-SEND-DATA	Used to transmit data to other members of a domain in a uniformly sequenced manner, i.e., the data will be received in the same sequence by all members of the destination channel. The different data units from the domain clients will be forwarded to the top MCS provider, which will send them back to all clients in the same sequence. Uniform sequencing of data is guaranteed only for data of the same priority on the same channel.
MCS-TOKEN-GRAB	Used to take exclusive control of a specific token.
MCS-TOKEN-INHIBIT	Used to take non-exclusive control of a specific token.

Table 6-1 – MCS primitives needed by an SICE

MCS primitive	Description
MCS-TOKEN-RELEASE	Used to free up a previously grabbed or inhibited token.
MCS-TOKEN-TEST	Used to check if a token is available.
MCS-TOKEN-PLEASE	Used to request a token held by another node.
MCS-TOKEN-GIVE	Used to pass a token directly from one node to another.

MCS request and response primitives are directed from the SICE to the MCS provider, while indication and confirm primitives are directed from the MCS provider towards the SICE. Additional detail on the MCS primitives described above can be found in [ITU-T T.122].

6.2 Use of MCS tokens and channels

Table 6-2 describes MCS channel and token usage for SICE sessions of the types defined in [ITU-T T.121]. In the case of a session type requiring static channels and tokens, the channel and token IDs shown in Table 6-2 shall be used (symbolic IDs shown). For all other session types, the resource IDs shown in the table shall be used for allocating dynamic tokens and channels. The given resource IDs shall be encoded as two-octet T.50 text strings using the characters shown in quotes in Table 6-2.

Table 6-2 – Description of SI tokens and channels

Mnemonic	Mnemonic for static channel and token IDs	Resource IDs for dynamic channels and tokens	Description
SI-{MCS-USER-ID}-CHANNEL	–	–	Certain SIPDUs are sent directly to individual SICEs. To do this, the individual MCS-USER-ID channels of the peer SICEs in the MCS domain are used.
SI-CHANNEL	SI-CHANNEL-0	"C0"	This channel bears all SIPDUs to be broadcast to all peer SICEs in a domain.
SI-BITMAP-CREATE-TOKEN	SI-TOKEN-0	"T0"	This token is used to restrict bitmap creation to allow only a single bitmap creation at a time. This token shall always be used for bitmaps destined to hard-copy workspaces. For soft-copy workspaces, this token shall only be used if the Soft-Copy-Bitmap-No-Token-Protection capability is not present in the negotiated capability set.

Table 6-2 – Description of SI tokens and channels

Mnemonic	Mnemonic for static channel and token IDs	Resource IDs for dynamic channels and tokens	Description
SI-WORKSPACE-REFRESH-TOKEN	SI-TOKEN-1	"T1"	This token is used to allow a specific SICE to become the designated workspace refresher. The SICE that holds this token is responsible for refreshing workspaces when a new peer SICE enrols into the session.

6.3 Use of MCS data services

Table 6-3 lists the use of the MCS data services MCS-SEND-DATA and MCS-UNIFORM-SEND-DATA for each SIPDU. This table includes the channel over which the data is sent, which of the two MCS primitives shall be used for the case of synchronized and unsynchronized workspaces, and the data priority at which the data is sent. If more than one channel priority is mandated, the SIPDU must be sent on all.

All PDUs specified in this Recommendation are placed in the data parameter of the MCS-SEND-DATA and MCS-UNIFORM-SEND-DATA primitives. The ASN.1-encoded PDUs are packed into the sequence of octets that form the data parameter such that the leading bit is placed in the most significant bit of each octet and filled toward the least significant bit of the octet.

Table 6-3 – Use of MCS data primitives for SIPDUs

SIPDU	Channel	MCS data primitive		Priority
		Synchronized workspaces	Unsynchronized workspaces (or hard-copy exchanges for valid SIPDUs) or exchanges involving single SICE access-protected workspace planes (Note 1)	
ArchiveAcknowledgePDU	User ID channel of source of ArchiveOpenPDU	MCS-SEND-DATA		Low
ArchiveClosePDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA		High, Medium, Low
ArchiveErrorPDU	User ID channel of source of archive command	MCS-SEND-DATA		Low
ArchiveOpenPDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA		Low

Table 6-3 – Use of MCS data primitives for SIPDUs

SIPDU	Channel	MCS data primitive		Priority
		Synchronized workspaces	Unsynchronized workspaces (or hard-copy exchanges for valid SIPDUs) or exchanges involving single SICE access-protected workspace planes (Note 1)	
BitmapAbortPDU	User ID channel of source of BitmapCreatePDU for request or SI-CHANNEL for announcement	MCS-SEND-DATA for request or MCS-UNIFORM-SEND-DATA for announcement	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	(Note 3)
BitmapCheckpointPDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	High
BitmapCreatePDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	(Note 3)
BitmapCreateContinuePDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	(Note 3)
BitmapDeletePDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	(Note 3)
BitmapEditPDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	(Note 3)
ConductorPrivilegeGrantPDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA		High
ConductorPrivilegeRequestPDU	User ID channel of the SICE at the conducting node	MCS-SEND-DATA		High
DrawingCreatePDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SD-DATA	Medium
DrawingDeletePDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	Medium
DrawingEditPDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	Medium
FontPDU	*FFS*	*FFS*		*FFS*
RemoteEventPermissionGrantPDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA		High

Table 6-3 – Use of MCS data primitives for SIPDUs

SIPDU	Channel	MCS data primitive		Priority
		Synchronized workspaces	Unsynchronized workspaces (or hard-copy exchanges for valid SIPDUs) or exchanges involving single SICE access-protected workspace planes (Note 1)	
RemoteEventPermissionRequestPDU	User ID channel of source of WorkspaceCreatePDU	MCS-SEND-DATA		High
RemoteKeyboardEventPDU	User ID channel of source of WorkspaceCreatePDU	MCS-SEND-DATA		High
RemotePointingDeviceEventPDU	User ID channel of source of WorkspaceCreatePDU	MCS-SEND-DATA		High
RemotePrintPDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	High (unsync'ed) High, Medium, Low (sync'ed)
SINonStandardPDU	(Note 2)	(Note 2)	(Note 2)	(Note 2)
TextCreatePDU	*FFS*	*FFS*		*FFS*
TextDeletePDU	*FFS*	*FFS*		*FFS*
TextEditPDU	*FFS*	*FFS*		*FFS*
VideoWindowCreatePDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	(Note 3)
VideoWindowDeletePDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	(Note 3)
VideoWindowEditPDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	(Note 3)
WorkspaceCreatePDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA		High (unsync'ed) High, Medium, Low (sync'ed)
WorkspaceCreateAcknowledgePDU	User ID channel of source of WorkspaceCreatePDU	–	MCS-SEND-DATA	High
WorkspaceDeletePDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	High (unsync'ed) High, Medium, Low (sync'ed)

Table 6-3 – Use of MCS data primitives for SIPDUs

SIPDU	Channel	MCS data primitive		Priority
		Synchronized workspaces	Unsynchronized workspaces (or hard-copy exchanges for valid SIPDUs) or exchanges involving single SICE access-protected workspace planes (Note 1)	
WorkspaceEditPDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	High (unsync'ed) High, Medium, Low (sync'ed)
WorkspacePlaneCopyPDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA	MCS-SEND-DATA or MCS-UNIFORM-SEND-DATA	(Note 3)
WorkspaceReadyPDU	SI-CHANNEL	–	MCS-SEND-DATA	High
WorkspaceRefreshStatusPDU	SI-CHANNEL	MCS-UNIFORM-SEND-DATA		High

NOTE 1 – MCS-SEND-DATA shall not be used unless the protectedPlaneAccessList includes only one SICE, the destination plane of the operation is designated protected and the SICE guarantees to never change the plane protection of the plane to unprotected.

NOTE 2 – The use of the SINonStandardPDU is beyond the scope of this Recommendation.

NOTE 3 – This indicates that the priority shall be medium if the destination plane has the annotation usage designator set or, in the case of bitmap operations, if the bitmap destination is the pointer plane, and low otherwise.

7 Use of GCC

7.1 Use of GCC services

An SICE uses the services of GCC [ITU-T T.124] in the manner specified in [ITU-T T.121]. The use of GCC by an SICE shall comply with the procedures outlined in [ITU-T T.121] in addition to the procedures explicitly described in this Recommendation.

7.2 GCC unique handles

All handles used in the SI protocol are acquired from GCC using the GCC-Registry-Allocate-Handle primitive. Handles can be allocated at any time, not just immediately prior to their use. It is suggested that applications allocate blocks of handles to minimize network traffic and perform this operation when idle to avoid associated latencies during periods of protocol activity.

8 Protocol specification

8.1 Session initialization and management

Session initialization and management for this Recommendation shall be performed according to [ITU-T T.121]. The functions of the application resource manager (ARM) described in [ITU-T T.121] shall apply to any T.126 session. For this Recommendation, the following initialization parameters shall be used. For the application protocol key, the value {ITU-T recommendation t126 version(0) 2} shall be used. The required channel and token

resources are defined in Table 6-2. The numeric values of the static channel and token IDs are specified in [ITU-T T.120].

In order to enable SICEs to perform session initialization functions such as (but not limited to) creating an initial whiteboard or displaying an introductory image, it is necessary for session members to be able to identify whether or not they are the first session participant upon entering the session. Identification of the first session participant shall be accomplished by employing a single GCC registry parameter. Upon active enrolment in a session, an SICE shall perform a GCC-Registry-Retrieve-Entry exchange with the resource ID portion of the registry key parameter set to the T.50-encoded octet string "INITIAL". If the result of the exchange reported by the GCC-Registry-Retrieve-Entry confirm is "successful", then the SICE shall deem itself not the initial session participant and not perform any startup operations such as creating an initial workspace. If the result is "entry not found", then the SICE shall attempt to create the same parameter using the GCC-Registry-Set-Parameter exchange using the same resource ID for the registry key and specifying "owner" for the optional modification rights parameter. If the result of this exchange is "successful", then the SICE has identified that it is the initial session participant and can perform its initial actions. If the result is "invalid requester", then the SICE shall not consider itself the first session participant. The scenario that could result in the later condition is one where multiple SICEs are simultaneously enrolling in a session.

8.2 Interpretation of optional parameters

It is mandatory that any PDU parameter or sub-parameter that is specified as optional in the descriptive text or ASN.1 of this Recommendation be appropriately interpreted and acted on by a receiving SICE unless otherwise stated. That is, it is mandatory for an SICE to be able to receive and interpret all optional parameters (unless otherwise stated) but all SICEs have the option of not sourcing optional parameters. If an optional parameter is left unspecified, any implied default value or behaviour defined in this Recommendation must also be adhered to. Some optional parameters may require successful negotiation of one or more SI capabilities before they may be used.

8.3 SI capabilities

Capabilities exchange and negotiation shall be performed according to [ITU-T T.121]. The SI capabilities that can be advertized and negotiated are described in Table 8-1. At least one of the following capabilities shall be included in the advertized application capabilities list: Hard-Copy-Image or Soft-Copy-Workspace. If Soft-Copy-Workspace is included, then at least one of the following shall also be included: Soft-Copy-Image or Soft-Copy-Annotation.

In Table 8-1, certain capabilities are indicated as being dependent on other capabilities. This implies that the capability shall not be included in the application capabilities list unless the capability on which it depends is also included.

SICEs are made aware of the application capabilities that are valid for a given session via GCC-Application-Roster-Report indications. The conditions under which this event is generated are described in [ITU-T T.121]. An SICE may be required to process GCC-Application-Roster-Report indications multiple times during a session and shall adhere to the bounds imposed by the capabilities reported in this fashion. Many components of this Recommendation heavily depend on this mechanism for proper operation, clauses within this Recommendation describing such components include rules governing the interpretation of the applicable parameters conveyed by the GCC-Application-Roster-Report indication.

Table 8-1 – Application capabilities list elements

Capability name (default value if numeric class): Description	ID	Class	SICE count rule	Parameter	Dependency
Hard-Copy-Image: This capability is used to negotiate the use of hard-copy image exchanges. This capability implies a maximum image size of 1728 horizontal by 2300 vertical. It also implies the ability to support unscaled image bitmap creation using either uncompressed or T.4 (G3) formats with a single bitplane and either fax1 or fax2 pixel aspect ratios.	1	L	>1	–	–
Hard-Copy-Image-Bitmap-Max-Width (1728): This capability is used to negotiate the maximum width of an image bitmap for hard-copy image exchanges. This dimension is relative to the pixel aspect ratio of the image.	2	MIN	=D	(1729..21845)	Hard-Copy-Image
Hard-Copy-Image-Bitmap-Max-Height (2300): This capability is used to negotiate the maximum height of an image bitmap for hard-copy image exchanges. This dimension is relative to the pixel aspect ratio of the image.	3	MIN	=D	(2301..21845)	Hard-Copy-Image
Hard-Copy-Image-Bitmap-Any-Aspect-Ratio: This capability is used to negotiate the ability to transmit image bitmaps to a hard-copy workspace with an arbitrary aspect ratio.	4	L	=D	–	Hard-Copy-Image
Hard-Copy-Image-Bitmap-Format-T.6: This capability is used to negotiate the ability to support bitmap creation using T.6 (G4) image compression format with a single bitplane and either fax1 or fax2 pixel aspect ratios.	5	L	=D	–	Hard-Copy-Image
Hard-Copy-Image-Bitmap-Format-T.82: This capability is used to negotiate the ability to support bitmap creation using T.82 (JBIG) image compression format. This capability implies the ability to handle 1 bitplane with 1:1 pixel aspect ratio and the ability to only handle bitmaps encoded without the use of JBIG resolution reduction.	6	L	=D	–	Hard-Copy-Image
Soft-Copy-Workspace: This capability is used to negotiate the ability to support at least one workspace for soft-copy information. This capability implies a maximum workspace size of 384 horizontal by 288 vertical with workspace background colours black and white. Presence of this capability also implies that one of the capabilities Soft-Copy-Annotation or Soft-Copy-Image shall also be included in the application capabilities list.	7	L	>1	–	–
Soft-Copy-Workspace-Max-Width (384): This capability is used to negotiate the maximum workspace width. This dimension is relative to a 1:1 pixel aspect ratio (square pixels).	8	MIN	=D	(385..21845)	Soft-Copy-Workspace

Table 8-1 – Application capabilities list elements

Capability name (default value if numeric class): Description	ID	Class	SICE count rule	Parameter	Dependency
Soft-Copy-Workspace-Max-Height (288): This capability is used to negotiate the maximum workspace height. This dimension is relative to a 1:1 pixel aspect ratio (square pixels).	9	MIN	=D	(289..21845)	Soft-Copy-Workspace
Soft-Copy-Workspace-Max-Planes (1): This capability is used to negotiate the maximum number of planes allowed in any workspace.	10	MIN	=D	(2..256)	Soft-Copy-Workspace
Soft-Copy-Color-16: This capability is used to negotiate the use of the 16-colour palette for use in workspace backgrounds or, if the Soft-Copy-Annotation capability is negotiated, in drawing elements.	11	L	=D	–	Soft-Copy-Workspace
Soft-Copy-Color-202: This capability is used to negotiate the use of the 202-colour palette for use in workspace backgrounds or, if the Soft-Copy-Annotation capability is negotiated, in drawing elements.	12	L	=D	–	Soft-Copy-Workspace
Soft-Copy-Color-True: This capability is used to negotiate the use of the true colour (24-bit RGB) as well as the use of the 202-colour palette for use in workspace backgrounds or, if the Soft-Copy-Annotation capability is negotiated, in drawing elements.	13	L	=D	–	Soft-Copy-Workspace
Soft-Copy-Plane-Editing: This capability is used to negotiate the ability to declare any workspace plane to be editable.	14	L	=D	–	Soft-Copy-Workspace
Soft-Copy-Scaling: This capability is used to negotiate the ability to declare a scaling rectangle during creation of soft-copy bitmaps and video windows. Without this capability, bitmaps and video windows are applied to the destination workspace without scaling (other than that required for non-1:1 pixel aspect ratios).	15	L	=D	–	Soft-Copy-Workspace
Soft-Copy-Bitmap-No-Token-Protection: This capability is used to negotiate the ability to transmit soft-copy bitmaps of any variety without the need to hold the SI-BITMAP-CREATE-TOKEN.	16	L	=D	–	Soft-Copy-Workspace
Soft-Copy-Pointing: This capability is used to negotiate the use of pointer bitmaps on soft-copy workspaces. Successful negotiation of this capability allows the following coding formats and associated parameter constraints for pointer bitmaps: – Uncompressed format of either 8-bit greyscale, RGB 4:4:4 or 1, 4 or 8-bit palettized with a 1:1 pixel aspect ratio. This capability implies the ability to handle pointer bitmaps up to a maximum size of 32 by 32 pixels.	17	L	>1	–	Soft-Copy-Workspace

Table 8-1 – Application capabilities list elements

Capability name (default value if numeric class): Description	ID	Class	SICE count rule	Parameter	Dependency
Soft-Copy-Pointing-Bitmap-Max-Width (32): This capability is used to negotiate the maximum width of a pointer bitmap. This dimension is relative to a 1:1 pixel aspect ratio (square pixels).	18	MIN	=D	(33..21845)	Soft-Copy-Pointing
Soft-Copy-Pointing-Bitmap-Max-Height (32): This capability is used to negotiate the maximum height of a pointer bitmap. This dimension is relative to a 1:1 pixel aspect ratio (square pixels).	19	MIN	=D	(33..21845)	Soft-Copy-Pointing
Soft-Copy-Pointing-Bitmap-Format-T.82: This capability is used to negotiate the ability to use T.82 (JBIG) compression format for encoding pointer bitmaps. This capability implies the ability to handle either 8-bit greyscale, or up to 8 palettized bitplanes with a 1:1 pixel aspect ratio and the ability to only handle bitmaps encoded without the use of JBIG resolution reduction.	20	L	=D	–	Soft-Copy-Pointing
Soft-Copy-Annotation: This capability is used to negotiate the use of annotation on soft-copy workspaces. The presence of this capability in the negotiated capability set implies the ability to create workspaces with annotation specified as the usage-designator of workspace planes. Successful negotiation of this capability also allows the following coding formats and associated parameter constraints for annotation bitmaps: – Uncompressed bitmap format of either 8-bit greyscale, RGB 4:4:4 or 1, 4, or 8-bit palettized raster and colour formats with a 1:1 pixel aspect ratio. This capability also implies the ability to support the creation of drawings using the DrawingCreatePDU with a pen thickness of 3 to 16 pixels, and a round pen nib.	21	L	>1	–	Soft-Copy-Workspace
Soft-Copy-Annotation-Bitmap-Max-Width (384): This capability is used to negotiate the maximum width of an annotation bitmap. This dimension is relative to a 1:1 pixel aspect ratio (square pixels).	22	MIN	=D	(385..65536)	Soft-Copy-Annotation
Soft-Copy-Annotation-Bitmap-Max-Height (288): This capability is used to negotiate the maximum height of an annotation bitmap. This dimension is relative to a 1:1 pixel aspect ratio (square pixels).	23	MIN	=D	(289..65536)	Soft-Copy-Annotation
Soft-Copy-Annotation-Drawing-Pen-Min-Thickness (3): This capability is used to negotiate the minimum thickness in pixels of lines drawn using the DrawingCreatePDU.	24	MAX	=D	(1..2)	Soft-Copy-Annotation

Table 8-1 – Application capabilities list elements

Capability name (default value if numeric class): Description	ID	Class	SICE count rule	Parameter	Dependency
Soft-Copy-Annotation-Drawing-Pen-Max-Thickness (16): The capability is used to negotiate the maximum thickness in pixels of lines drawn using the DrawingCreatePDU.	25	MIN	=D	(17..255)	Soft-Copy-Annotation
Soft-Copy-Annotation-Drawing-Ellipse: This capability is used to negotiate the ability to use the ellipse drawing type when using the DrawingCreatePDU.	26	L	=D	–	Soft-Copy-Annotation
Soft-Copy-Annotation-Drawing-Pen-Square-Nib: This capability is used to negotiate the ability to use a square nib shape in creation of lines drawn using the DrawingCreatePDU.	27	L	=D	–	Soft-Copy-Annotation
Soft-Copy-Annotation-Drawing-Highlight: This capability is used to negotiate the ability to make use of the Highlight line style for drawing.	28	L	=D	–	Soft-Copy-Annotation
Soft-Copy-Annotation-Drawing-Rotation: This capability is used to negotiate the ability to specify the optional rotation parameter that defines a rotation to be applied to annotation drawing elements.	62	L	=D	–	Soft-Copy-Annotation
Soft-Copy-Annotation-Bitmap-Format-T.82: This capability is used to negotiate the ability to use T.82 (JBIG) compression format for encoding annotation bitmaps. This capability implies the ability to handle either 8-bit greyscale, or up to 8 palletized bitplanes with a 1:1 pixel aspect ratio and the ability to only handle bitmaps encoded without the use of JBIG resolution reduction.	29	L	=D	–	Soft-Copy-Annotation

Table 8-1 – Application capabilities list elements

Capability name (default value if numeric class): Description	ID	Class	SICE count rule	Parameter	Dependency
Soft-Copy-Image: This capability is used to negotiate the use of image bitmaps on soft-copy workspaces. The presence of this capability in the negotiated capability set implies the ability to create workspaces with image specified as the usage-designator of workspace planes. Successful negotiation of this capability allows the following coding formats and associated parameter constraints for image bitmaps: 1) JBIG: This capability implies the ability to handle either 8-bit greyscale, RGB 4:4:4, or up to 8 palettized bitplanes and the ability to only handle bitmaps encoded without the use of JBIG resolution reduction. Both 1:1 and CIF pixel aspect ratios shall be supported. 2) JPEG: This capability implies the ability to handle the baseline DCT encoding mode, with baseline sequential transmission and 8 bit/sample data precision in component interleaved format only, using a colour space and colour resolution mode of YCbCr 4:2:2, or greyscale. Both 1:1 and CIF pixel aspect ratios shall be supported. 3) Uncompressed: This capability implies the ability to handle 8-bit greyscale, RGB 4:4:4, YCbCr 4:2:2, or palettized 1, 4, or 8 bits per pixel. Both 1:1 and CIF pixel aspect ratios shall be supported.	30	L	>1	–	Soft-Copy-Workspace
Soft-Copy-Image-Bitmap-Max-Width (384): This capability is used to negotiate the maximum workspace width for soft-copy image bitmap exchanges. This dimension is relative to the pixel aspect ratio of the image bitmap.	31	MIN	=D	(385..65536)	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Max-Height (288): This capability is used to negotiate the maximum workspace height for soft-copy image bitmap exchanges. This dimension is relative to the pixel aspect ratio of the image bitmap.	32	MIN	=D	(289..65536)	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Any-Aspect-Ratio: This capability is used to negotiate the ability to transmit image bitmaps to a soft-copy workspace with an arbitrary aspect ratio.	33	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.82-Differential: This capability is used to negotiate the ability to use resolution reduction (differential layers) when encoding a JBIG format image bitmap.	34	L	=D	–	Soft-Copy-Image

Table 8-1 – Application capabilities list elements

Capability name (default value if numeric class): Description	ID	Class	SICE count rule	Parameter	Dependency
Soft-Copy-Image-Bitmap-Format-T.82-Differential-Deterministic-Prediction: This capability is used to negotiate the ability to use deterministic prediction when encoding a JBIG format image bitmap using resolution reduction (differential layers).	35	L	=D	–	Soft-Copy-Image-Bitmap-Format-T.82-Differential
Soft-Copy-Image-Bitmap-Format-T.82-12-Bit-Grey-Scale: This capability is used to negotiate the ability to use 12 bitplanes when encoding a JBIG format image bitmap.	36	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Extended-Sequential-DCT: This capability is used to negotiate the ability to use extended sequential DCT mode when encoding a JPEG format image bitmap.	37	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Progressive-DCT: This capability is used to negotiate the ability to use progressive DCT mode when encoding a JPEG format image bitmap.	38	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Spatial-DPCM: This capability is used to negotiate the ability to use spatial DPCM mode when encoding a JPEG format image bitmap.	39	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Sequential-DCT: This capability is used to negotiate the ability to use differential sequential DCT mode when encoding a JPEG format image bitmap.	40	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Progressive-DCT: This capability is used to negotiate the ability to use differential progressive DCT mode when encoding a JPEG format image bitmap.	41	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Spatial-DPCM: This capability is used to negotiate the ability to use differential spatial DPCM mode when encoding a JPEG format image bitmap.	42	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Extended-Sequential-DCT-Arithmetic: This capability is used to negotiate the ability to use extended sequential DCT mode using arithmetic encoding when encoding a JPEG format image bitmap.	43	L	=D	–	Soft-Copy-Image

Table 8-1 – Application capabilities list elements

Capability name (default value if numeric class): Description	ID	Class	SICE count rule	Parameter	Dependency
Soft-Copy-Image-Bitmap-Format-T.81-Progressive-DCT-Arithmetic: This capability is used to negotiate the ability to use progressive DCT mode using arithmetic encoding when encoding a JPEG format image bitmap.	44	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Spatial-DPCM-Arithmetic: This capability is used to negotiate the ability to use spatial DPCM mode using arithmetic encoding when encoding a JPEG format image bitmap.	45	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Sequential-DCT-Arithmetic: This capability is used to negotiate the ability to use differential sequential DCT mode using arithmetic encoding when encoding a JPEG format image bitmap.	46	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Progressive-DCT-Arithmetic: This capability is used to negotiate the ability to use differential progressive DCT mode using arithmetic encoding when encoding a JPEG format image bitmap.	47	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Spatial-DPCM-Arithmetic: This capability is used to negotiate the ability to use differential spatial DPCM mode using arithmetic encoding when encoding a JPEG format image bitmap.	48	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-YCbCr-4:2:0: This capability is used to negotiate the ability to use a chroma format of YCbCr 4:2:0 when encoding a JPEG format image bitmap.	49	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-YCbCr-4:4:4: This capability is used to negotiate the ability to use a chroma format of YCbCr 4:4:4 when encoding a JPEG format image bitmap.	50	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-RGB-4:4:4: This capability is used to negotiate the ability to use a chroma format of RGB 4:4:4 when encoding a JPEG format image bitmap.	51	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-CIELab-4:2:0: This capability is used to negotiate the ability to use a chroma format of CIELab 4:2:0 when encoding a JPEG format image bitmap.	52	L	=D	–	Soft-Copy-Image

Table 8-1 – Application capabilities list elements

Capability name (default value if numeric class): Description	ID	Class	SICE count rule	Parameter	Dependency
Soft-Copy-Image-Bitmap-Format-T.81-CIELab-4:2:2: This capability is used to negotiate the ability to use a chroma format of CIELab 4:2:2 when encoding a JPEG format image bitmap.	53	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-CIELab-4:4:4: This capability is used to negotiate the ability to use a chroma format of CIELab 4:4:4 when encoding a JPEG format image bitmap.	54	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-T.81-Non-Interleaved: This capability is used to negotiate the ability to use non-interleaved ordering of colour components.	55	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-Uncompressed-YCbCr-4:2:0: This capability is used to negotiate the ability to use a chroma format of YCbCr 4:2:0 when encoding an *uncompressed* format image bitmap.	56	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-Uncompressed-YCbCr-4:4:4: This capability is used to negotiate the ability to use a chroma format of YCbCr 4:4:4 when encoding an *uncompressed* format image bitmap.	57	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-Uncompressed-CIELab-4:2:0: This capability is used to negotiate the ability to use a chroma format of CIELab 4:2:0 when encoding an *uncompressed* format image bitmap.	58	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-Uncompressed-CIELab-4:2:2: This capability is used to negotiate the ability to use a chroma format of CIELab 4:2:2 when encoding an *uncompressed* format image bitmap.	59	L	=D	–	Soft-Copy-Image
Soft-Copy-Image-Bitmap-Format-Uncompressed-CIELab-4:4:4: This capability is used to negotiate the ability to use a chroma format of CIELab 4:4:4 when encoding an *uncompressed* format image bitmap.	60	L	=D	–	Soft-Copy-Image
Archive-Support: This capability is used to negotiate the support of archives.	61	L	–	–	–

Table 8-1 – Application capabilities list elements

Capability name (default value if numeric class): Description	ID	Class	SICE count rule	Parameter	Dependency
Soft-Copy-Transparency-Mask: This capability is used to negotiate the ability to use arbitrary transparency masks for applicable graphical elements allowing arbitrary pixels within these objects to be interpreted as transparent. This capability also implies the support of JBIG compression given that a transparency mask can be optionally encoded in this manner.	63	L	>1	–	Soft-Copy-Image
Soft-Copy-Video-Window: This capability is used to negotiate the ability to define video windows that can encapsulate out-of-band video streams in a workspace. Successful negotiation of this capability between two or more session participants enables the use of the VideoWindowCreatePDU, VideoWindowDeletePDU and VideoWindowEditPDU.	64	L	>1	–	Soft-Copy-Image
Non-Standard-Capability: This capability is used to negotiate non-standard functions. Any number of these may appear in the application capabilities list as long as they each have a unique Non-Standard-Identifier. The interpretation of these capabilities is not covered by this Recommendation.	Non-Standard Identifier	–	–	–	–

Table 8-2 – Capability list notation

Class	L	Logical
	MIN	Unsigned minimum
	MAX	Unsigned maximum
SICE count rule	>1	The Number-of-Entities parameter returned by the GCC-Application-Roster-Report indication for this capability must be greater than 1 for it to be considered established for the session.
	=D	In the case of logical capabilities, the Number-of-Nodes parameter returned by the GCC-Application-Roster-Report indication for this capability must be equal to that of the dependent capability for it to be considered established. In the case of numerical capabilities (MIN or MAX), if the Number-of-Nodes parameter is equal to that of the dependent capability, the result of the MIN or MAX operation is established as the session-wide value; otherwise the default value of the capability is established.

8.4 Workspaces

A workspace may be created if the Soft-Copy-Workspace capability is present in the negotiated capability set. Multiple workspaces may coexist if sufficient resources are present at all peer SICEs within a session. This facility can be useful for multi-document applications as well as serving as a

method to cache frequently used graphical information to help avoid the delays that would otherwise be incurred if the information had to be sent at the time it was used in the session.

Workspaces may be created and deleted at any time with different attributes during a session. New arrivers to the session will be able to view information upon receipt of the first workspace create exchange they receive. Workspace attributes within a multi-workspace SI session can be different from one workspace to another. No attributes shall exceed any negotiated limits.

8.4.1 Workspace structure

8.4.1.1 Workspace plane stacking

A workspace consists of a number of stacked planes which aid in determining how overlapping graphical objects occlude each other. The number of planes in a workspace is defined at the time of creation and may range from 1 to the maximum value negotiated for the session using the Soft-Copy-Workspace-Max-Planes capability. An additional virtual plane used exclusively for pointers is present if the Soft-Copy-Pointing capability has been negotiated for the session.

For a workspace with N planes, the planes are ordered from 0 to N – 1 with 0 being treated as the farthest back and N – 1 being the topmost. If the Soft-Copy-Pointing capability has been successfully negotiated, the pointer plane forms a virtual plane in front of all other planes.

When rendering an image to be displayed, the colour value of each pixel (prior to any format conversion necessary for display) is determined by the following set of rules:

- If a plane X contains a non-transparent graphical element at the topmost Z-order for elements within that plane at that workspace coordinate, and all planes farther forward (including the virtual pointer plane) contain transparent pixel values at that same location, then the resulting pixel shall be set to the colour of the corresponding pixel on plane X.
- If the pixel value for all planes 0 through N (includes virtual pointer plane) are set to transparent, then the resulting pixel shall be set to the background colour specified when the workspace was created.
- If plane X contains a drawing element specified with a line style of highlight, and all planes farther forward (from X + 1 through N), if any, contain transparent pixel values, then a locally-defined rule is applied to modify the pixel value that would result if these rules were applied only to planes 0 through X – 1 (X is equal to 0, the modification rule is applied to the background colour). The locally-defined modification rule should have the effect of modifying the image formed by rending the layers below X in such a way that it appears that the resulting pixel value has been highlighted with a semi-transparent coloured value of the colour specified at this pixel in plane X. The specific algorithm for this rule is outside of the scope of this Recommendation.

8.4.1.2 Workspace plane coordinate system

All planes on a workspace are of the same size with the origin of each co-located. Within each plane, pixels are indexed from the origin (0,0), which is defined to be the upper left corner of the workspace, to the size of the workspace (X – 1, Y – 1) which is defined to be the lower right corner of the workspace, where X and Y are the number of pixels in the workspace, in the horizontal and vertical dimensions, respectively, as specified when the workspace was created. The sizes X and Y shall be greater than or equal to one, and less than or equal to the negotiated maximum values of Soft-Copy-Workspace-Max-Width and Soft-Copy-Workspace-Max-Height, respectively.

Positional references to a workspace plane are designated by specifying a point. A point is an ordered pair of workspace coordinates specifying the horizontal and vertical position in the workspace, respectively. The value of a workspace coordinate is defined to be within the range (–21845..43690). The use of negative values allows objects (e.g., the origin of a bitmap, or a control

point in a drawing element) to be positioned above or to the left of the origin of the workspace plane.

All references to workspace size and plane coordinates assume 1:1 pixel aspect ratio (square pixels) regardless of the aspect ratio of bitmaps that may be transmitted to the workspace.

8.4.1.3 Workspace views

A workspace view defines a rectangular region of a workspace and its associated attributes that may be mapped to the display. For some or all of a workspace to be viewed by participants in a session, at least one workspace view must be defined for that workspace. A workspace view is a region of the workspace to be viewed and associated characteristics to describe how that region should be viewed. A single workspace may be defined to have up to 256 views. Each view may cover distinct or overlapping regions of the workspace, or some views may show common portions of the workspace.

Only active workspaces (non-archive) may be defined to have views – archived workspaces cannot. If views are defined for an archived workspace, they shall be ignored.

Views do not exist separately from the workspace to which they are associated. If a workspace is deleted, all of its associated views are automatically deleted as well.

Figure 8-1 shows the correspondence between a set of workspace views and the workspace to which they are associated.

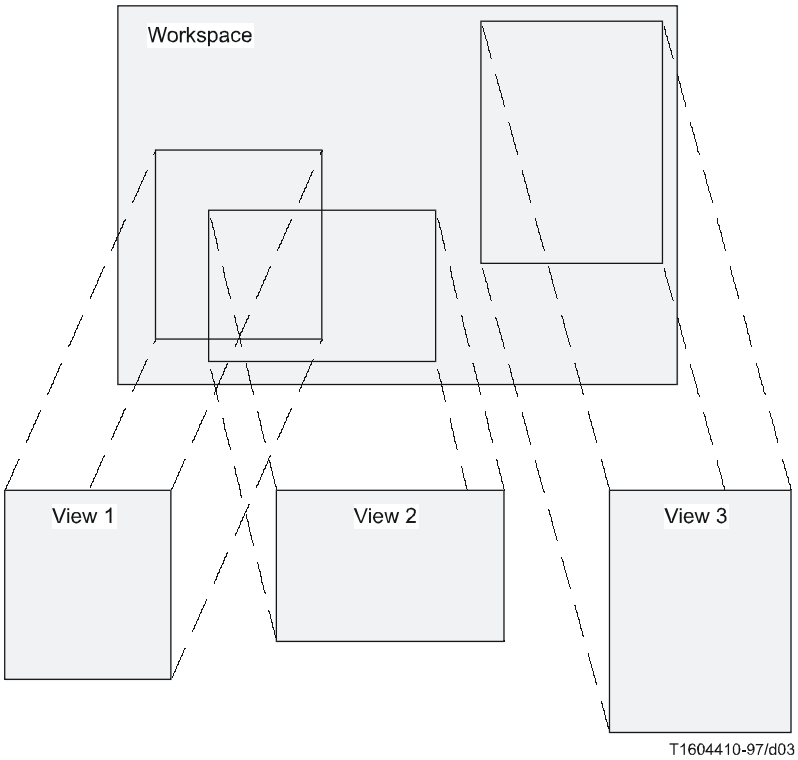

Figure 8-1 – Workspace views

8.4.2 Creating a workspace

To create a workspace, an SICE shall send a WorkspaceCreatePDU to all SICEs in the session in the manner indicated in Table 6-3. The content of the WorkspaceCreatePDU is shown in Table 8-3. If the workspace is synchronized, it shall send this SIPDU three separate times on each of the three priority channels, high, medium and low. This is done to avoid any possibility of data bound for this workspace on any priority channel being received prior to the WorkspaceCreatePDU itself. The first

one received shall be acted on by the receiving SICE. All others shall be ignored. In the case of an unsynchronized workspace, only a single WorkspaceCreatePDU is issued as indicated in Table 6-3.

Table 8-3 – WorkspaceCreatePDU

Parameter	Description
workspaceIdentifier	If this is an active (non-archived) workspace, this is a handle allocated using the GCC-Registry-Allocate-Handle primitive. If this is an archived workspace, this consists of the handle of the opened archive along with the entry name to be used to identify this workspace within the specified archive. Also included in this case is a modificationTime parameter indicating the current time. This shall not be an archived workspace if the Archive-Supported capability is not present in the negotiated capability set.
appRosterInstance	Application roster instance number as returned in the most recently received GCC-Application-Roster-Report indication.
synchronized	Either synchronized or unsynchronized. A synchronized workspace is one that guarantees that the workspace at each node in the session contains the same graphical information at the same Z-order. This is done by requiring synchronized transmission for all SIPDUs using the MCS-UNIFORM-SEND-DATA service for all exchanges that can result in different workspace contents if not received in the same order by all SICEs in a session. For an unsynchronized workspace, no guarantee of visual consistency is made due to potential timing conditions in the communications network. In the unsynchronized case, MCS-SEND-DATA is used for some SIPDUs rather than MCS-UNIFORM-SEND-DATA, resulting in a reduction in network traffic. The valid use of MCS-SEND-DATA versus MCS-UNIFORM-SEND-DATA for all SIPDUs is summarized in Table 6-3.
acceptKeyboardEvents	This flag expresses the ability of the workspace to accept keyboard events via the RemoteKeyboardEventPDU.
acceptPointingDeviceEvents	This flag expresses the ability of the workspace to accept pointing device events via the RemotePointingDeviceEventPDU.
protectedPlaneAccessList (optional)	An optional set of MCS user IDs. If present, the ability to modify any protected plane in this workspace is restricted only to SICEs in this set. The creator of the workspace is not automatically granted access to these planes unless its MCS user ID is specifically on this list (if the list is present). If the parameter is not specified, all SICEs are assumed to have access to the workspace planes.
workspaceSize	This parameter indicates the horizontal and vertical dimension of the workspace. This parameter is expressed in units of pixels. The horizontal component of this parameter may span the range (1..X) where X is the value of the Soft-Copy-Workspace-Max-Width parameter in the negotiated capability set. The vertical component of this parameter may span the range (1..Y) where Y is the value of the Soft-Copy-Workspace-Max-Height parameter in the negotiated capability set. The pixels used to define the workspace size are defined as having a 1:1 pixel aspect ratio (square pixels).
workspaceAttributes (optional)	Workspace attributes are described in Table 8-4. These can be later edited via the WorkspaceEditPDU.
planeParameters	Workspace plane parameters are described in Table 8-5.

Table 8-3 – WorkspaceCreatePDU

Parameter	Description
viewParameters (optional)	Workspace view parameters are described in Table 8-7. When a workspace is created, it may be created along with one or more workspace views.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.
refresh (optional)	A value of TRUE indicates that the WorkspaceCreatePDU is being used to initiate a workspace refresh. The value of FALSE, or the parameter not being present, indicates that the WorkspaceCreatePDU is being used to create a new workspace. See clause 8.4.7 for the rules for interpreting workspace create exchanges used to refresh late arrivers to a session.

Table 8-4 – Workspace attributes

Attribute	Default value	Description
preserve	FALSE	This is an optional parameter which indicates the preference to preserve this workspace (resources permitting). Its value has an impact on the behaviour of an SICE, should it exceed its resource limits. See clause 8.4.9 for details. This flag should only be set to TRUE on rare occasions. It is not intended for use when pre-transmitting SI workspaces for later use. Its primary function is to allow workspaces being used for graphical element caches to be preserved. The preserve flag may be modified by sending a WorkspaceEditPDU specifying a new value.
backgroundColor	White	backgroundColor is used to determine the resulting colour of a pixel if all plane contents (including the virtual pointer plane) are set to a transparent value at that pixel location. In the case of only objects with highlight line styles present above the workspace background, the pixel colour shall be the backgroundColor combined with the colour of the pixels in front using the highlighting rule. The colour space over which the background colour may be chosen is determined by the negotiated palette for the corresponding workspace type (see clause 8.4.6). The baseline palette is the two-colour palette. If a broader colour space was negotiated for use, the corresponding colour spaces may be used to set the background colour. Specifically, if the Soft-Copy-Color-16, Soft-Copy-Color-202 or Soft-Copy-Color-True capabilities are present in the negotiated capability set, then the 16-colour palette, the 202-colour palette or a true-colour specifier may be used, respectively. The background colour may be modified by sending a WorkspaceEditPDU specifying a new value.

Table 8-4 – Workspace attributes

Attribute	Default value	Description
nonStandardAttribute	–	This attribute is specified as a nonStandardIdentifier. To be used, it must have been successfully negotiated by a corresponding non-standard capability. Its interpretation is not specified by this Recommendation.

Table 8-5 – Workspace plane parameters

Parameter	Description
editable	This flag indicates whether the plane shall be permanent or editable. This flag shall be set to permanent unless the Soft-Copy-Workspace-Editing capability is present in the negotiated capability set, in which case it may be designated as either permanent or editable. For a plane designated as permanent, the effect of graphical object creation shall be to modify the image as a single-plane raster image. That is, each pixel in the plane has one and only one value that determines its colour (or transparency) and carries no other state information. Its value shall correspond to the one last written to that plane location as a result of the creation (including copying) of a new graphical element. Bitmaps and drawing elements on a permanent plane cannot be edited or deleted. They can be modified only by overwriting their pixel locations with other bitmaps or drawing elements. As a result, no graphical object edits or deletes can be directed to permanent planes. If an SICE receives any PDU attempting to modify the attributes or position of an object residing in a permanent plane, it shall be ignored. For a plane designated as editable, each SICE shall maintain a database of state information for each graphical element created on that plane. For each object, this shall include the information defining the object, its attributes, its position relative to the origin of the plane, and the position of the object in the object-list for that plane (the Z-order). The object-list determines the order of the objects within the plane, from front to back, used to determine which objects occlude others.

Table 8-5 – Workspace plane parameters

Parameter	Description
usage	This flag indicates the intended usage for the plane: for either image bitmaps, annotation (drawing elements or annotation bitmaps), nonStandardPlaneUsage or a combination of these. This parameter may include annotation only if the Soft-Copy-Annotation capability is present in the negotiated capability set. This parameter may include image only if the Soft-Copy-Image capability is present in the negotiated capability set. This parameter may include a nonStandardPlaneUsage only if a corresponding non-standard capability is present in the negotiated capability set. When a workspace is created, the intended use of each plane in the workspace shall be designated using the usage designator in the WorkspaceCreatePDU. The usage designator is a set of flags. Each member of the set can have three possible values. The first flag indicates whether the plane may be used for annotation. The second indicates whether the plane may be used for image data. The third is used to designate a nonStandardUsage (which requires successful negotiation of a corresponding nonStandardCapability for use). Multiple flags can be asserted for each plane but at least one of these flags must be set. The annotation flag shall not be set unless the Soft-Copy-Annotation capability has been negotiated for the session, and the image flag shall not be set unless the Soft-Copy-Image capability has been successfully negotiated for the session. For a plane designated as allowing annotation, the plane may be used for drawing elements or annotation bitmaps. Drawing elements are those created using the DrawingCreatePDU, while annotation bitmaps are those created using the BitmapCreatePDU with the destinationAddress parameter set to softCopyAnnotationPlane. For a plane designated as allowing image usage, the plane may be used for image bitmaps and video windows. Image bitmaps are those created using BitmapCreatePDU with the destinationAddress parameter set to softCopyImagePlane. The usage designator of each plane determines the data priority used to transmit any SIPDU directed toward that plane as specified in Table 6.3.
planeAttributes (optional)	A set of attributes for a plane (see Table 8-6).

Table 8-6 – Plane attributes

Attribute	Default value	Description
protection	FALSE	This flag indicates access restrictions for a plane. The protected sub-parameter, if set to TRUE, restricts all operations to the plane to be by the SICEs specified in the protectedPlaneAccessList of the workspace only. A value of FALSE imposes no restrictions. If a workspace plane is designated as unprotected, any SICE in a session may direct information toward the plane. Protected workspace planes automatically revert to unprotected when all the SICEs identified by the protectedPlaneAccessList for that workspace leave the session or become inactive as indicated by a GCC-Application-Roster-Report indication. Workspace planes shall not be designated protected for archive workspaces. All parameters associated with plane protection shall be ignored in the case of archive workspaces. If a workspace plane is designated as protected, only the SICEs specified in the workspace's protectedPlaneAccessList may direct information toward the plane. Moreover, if the protectedPlaneAccessList contains only a single SICE, that SICE can choose to use MCS-SEND-DATA instead of MCS-UNIFORM-SEND-DATA for transactions directed to a workspace plane when the destination workspace plane is synchronized, if that SICE never intends to unprotect the target plane. Only SICEs specified in the protectedPlaneAccessList may edit this attribute using the WorkspaceEditPDU. If a WorkspaceEditPDU is received from any node other than the SICEs specified in the protectedPlaneAccessList which indicates a change in the value of this attribute, this change shall be ignored (other attribute changes shall still be processed as normal).
nonStandardAttribute	–	This attribute is specified as a nonStandardIdentifier. To be used, it must have been successfully negotiated by a corresponding non-standard capability. Its interpretation is not specified by this Recommendation.

Table 8-7 – Workspace view parameters

Parameter	Description
viewHandle	Unique handle returned from GCC-Registry-Allocate-Handle exchange. This handle will be used to reference this view in all future SIPDUs.
viewAttributes (optional)	A set of attributes for a view (see Table 8-8).

Table 8-8 – View attributes

Attribute	Default value	Description
viewRegion	fullWorkspace	This attribute defines the portion of the workspace to be associated with this view. This may be set to either fullWorkspace, which indicates that the view corresponds to the complete workspace, or partialWorkspace. In the later case, a rectangular region is specified, in workspace coordinates, indicating the desired portion of the workspace to view. If the viewRegion extends beyond the boundary of the workspace in any dimension, the displayed contents of those portions of the view that extend beyond the workspace boundary are not specified by this Recommendation. Each workspace view is defined to cover a rectangular region of the workspace to which it corresponds. The viewRegion parameter defines the size and location of this region. Its default value is to view the entire workspace. Alternatively, it may be set to view a particular sub-region of the workspace. If the viewRegion extends beyond the boundary of the corresponding workspace in any dimension, the displayed contents of those portions of the view that extend beyond the workspace boundary are not specified by this Recommendation.
viewState	focus	This attribute indicates the initial state of the view, either hidden, background, foreground, focus or nonStandardState. When a workspace view is created, the state of the view may be specified as having one of the following values: • Hidden – Indicates this view should not be shown to the user. • Background – Indicates that the display of this view is optional. • Foreground – Indicates that the display of this view is desirable. • Focus – Indicates that this view is the focus of the current session. The state of the view may be modified by sending a WorkspaceEditPDU specifying a new value of the viewState attribute for this view. Only one view among all views on all workspaces may simultaneously be set to the Focus state. Setting a view to this state forces any other view currently in the focus state into the foreground state.

Table 8-8 – View attributes

Attribute	Default value	Description
		An SICE may choose to send PDUs transmitted to workspaces in whose views are in different states at different throughput rates. For example, an SICE may send PDUs to workspaces whose views are background or hidden at a lower rate than PDUs to workspaces with foreground views or the workspace with a view in the focus state. This would minimize interference on information bound for the more important workspaces from information being sent in the background. The mechanism for controlling the throughput of PDUs intended for different workspaces is a local matter not covered by this Recommendation.
updatesEnabled	TRUE	This attribute specifies whether or not it is recommended that updates to the workspace associated with this view are displayed, or if the content of the view is instead fixed until updates are enabled. This attribute is to be interpreted as a recommendation on how to display the view rather than a requirement. If more than one SICE set this parameter to the disabled state (FALSE), an SICE which makes use of this attribute shall keep track of all such requesting SICEs and should only re-enable updates to the view when all of these SICEs edit the attribute back to the TRUE state (at which time views are re-enabled). If an SICE leaves the session after setting this attribute to FALSE, all other SICEs should interpret this as an implied re-editing of this attribute to TRUE (from that SICE only). It is recommended that updates not be disabled for more than short periods of time (e.g., only during bursts of rapid activity on the workspace).
sourceDisplayIndicator	–	This attribute specifies the size and position of the view relative to the display at the sourcing terminal. This is to be interpreted as a recommendation to the receiving SICE to display the view in a similar manner. However, there is no requirement that the receiving SICE actually display the view in the indicated manner. The creator of a view may specify the size and position of the view relative to the display at the creator's terminal. This is meant as a recommendation to receiving SICEs that this view be displayed in a similar manner. However, there is no requirement that the receiving terminal actually display the view in this way. The sourceDisplayIndicator is composed of the set of parameters indicated in Table 8-9.
nonStandardAttribute	–	This attribute is specified as a nonStandardIdentifier. To be used, it must have been successfully negotiated by a corresponding non-standard capability. Its interpretation is not specified by this Recommendation.

Table 8-9 – Source display indicator

Parameter	Description
displayAspectRatio	The aspect ratio of the display device being indicated. This is defined to be the ratio of horizontal to vertical dimensions of the display area.
horizontalSizeRatio	This is the ratio of the horizontal size of the view to the horizontal display size.
horizontalPosition	This is the horizontal offset of the upper left hand corner of the view relative to the upper left hand corner of the display. This is given in normalized units relative to a horizontal display width of 1.0.
verticalPosition	This is the vertical offset of the upper left hand corner of the view relative to the upper left hand corner of the display. This is given in normalized units relative to a vertical display height of 1.0.

Any node that receives a WorkspaceCreatePDU on any of the three priority channels shall first examine the synchronized flag in the SIPDU to determine if the workspace is designated as synchronized or unsynchronized. If the workspace is unsynchronized, the SICE shall send a WorkspaceCreateAcknowledgePDU to the node that issued the WorkspaceCreatePDU. This is done as indicated in Table 6-3. The content of the WorkspaceCreateAcknowledgePDU is shown in Table 8-10. An SICE shall not issue any exchanges that require the use of a priority channel that a WorkspaceCreatePDU has not been received on in the case of synchronized workspaces. Only a single WorkspaceCreatePDU is issued in the case of unsynchronized workspaces.

Table 8-10 – WorkspaceCreateAcknowledgePDU

Parameter	Description
workspaceIdentifier	This specifies the workspace being acknowledged. This may be either an active or archived workspace, although this shall not be an archived workspace if the Archive-Supported capability is not present in the negotiated capability set. In the case of an active workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace. In the case of an archived workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace except the modificationTime parameter shall not be included.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Next, the SICE shall determine how it should process the WorkspaceCreatePDU. First, the SICE shall compare the application roster instance number specified in the WorkspaceCreatePDU against the last application roster instance number that was received that signalled new SICEs entering the session. If the new application roster instance is less than this number, the SICE shall ignore the WorkspaceCreatePDU (and all WorkspaceCreatePDUs with the same workspace handle received later on other priority channels). If the application roster instance number specified in the WorkspaceCreatePDU is greater than or equal to the last application roster instance number that was received that signalled new SICEs entering the session and less than or equal to the last application roster instance number received, the WorkspaceCreatePDU is accepted and processed, creating the workspace with the characteristics designated by the parameters in the SIPDU. Because race conditions exist between the receipt of GCC-Application-Roster-Report indications and WorkspaceCreatePDUs, it is possible for the application roster instance number specified in the WorkspaceCreatePDU to be greater than the latest instance number reported by a GCC-Application-Roster-Report indication. In this case, the SICE receiving the

WorkspaceCreatePDU shall process in advance or buffer the workspace create transaction and any other valid transactions directed to the workspace until a GCC-Application-Roster-Report indication is received that identifies an instance number which is equal to the one specified in the workspace create. The SICE shall not source any exchanges to this workspace (except acknowledgements) until this time because the correct capabilities set that binds the new workspace is not known until the referenced GCC-Application-Roster-Report indication is received. If the SICE chooses to process any workspace transactions in advance of the receipt of the proper GCC-Application-Roster-Report indication, it must not assume that the capabilities associated with the new workspace are within the bounds defined by its own capability set given that the roster change prompting the new application roster may have been triggered by the SICE being either forcibly or voluntarily removed from the session.

An SICE that receives a WorkspaceCreatePDU before it receives its first GCC-Application-Roster-Report indication shall process in advance or buffer the workspace create transaction and any other valid transactions directed to the workspace until a GCC-Application-Roster-Report indication is received that identifies an instance number that is equal to the one specified in the workspace create. The same care must be taken with respect to the temporary capability ambiguity as described above. When the application roster does arrive, any workspaces whose application roster instance number is less than the instance number reported by the GCC-Application-Roster-Report indication shall be locally discarded.

As the SICE receives the remaining two WorkspaceCreatePDUs on each of the other two priority channels, it shall check that their workspace handle is the same as that of a WorkspaceCreatePDU previously received on another priority channel, and if so, it shall ignore it.

If more than one workspace is being created at the same time by different SICEs, it is possible that the three WorkspaceCreatePDUs from one SICE will be interleaved with the three from other SICEs (i.e., the order that they are received on each of the three priority channels may be different). If multiple WorkspaceCreatePDUs specify views with focus set as the viewState, assertion of focus is applied in the order that the WorkspaceCreatePDUs are received on the high priority channel.

If the workspace is designated as unsynchronized, the originating SICE shall wait to receive WorkspaceCreateAcknowledgePDUs from each of the SICEs which are present and marked as active in the instance of the application roster to which the workspace was referenced. If a new application roster is received in which some of the nodes in the original instance are no longer present, the SICE shall consider those nodes as having acknowledged and will no longer wait for a response from them. If the SICE receives an acknowledgment from any node which was not in the application roster instance to which the workspace was referenced, these acknowledgements shall be ignored. Once all acknowledgements have been received, the SICE creating the workspace shall issue a WorkspaceReadyPDU with the parameters set according to Table 6-3. This SICE can then perform all allowable exchanges to the new workspace. All other SICEs in the session shall wait to receive the WorkspaceReadyPDU before performing any exchanges to the new workspace. However, information from other SICEs directed at the workspace can be received prior to reception of the WorkspaceReadyPDU and should be considered valid.

There are additional rules and conditions regarding the processing of multiple WorkspaceCreatePDUs from multiple SICEs attempting to refresh the same workspace simultaneously. These can be found in clause 8.4.7.

Table 8-11 – WorkspaceReadyPDU

Parameter	Description
workspaceIdentifier	This specifies the workspace being identified as ready. This may be either an active or archived workspace, although this shall not be an archived workspace if the Archive-Supported capability is not present in the negotiated capability set. In the case of an active workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace. In the case of an archived workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace except the modificationTime parameter shall not be included.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.4.3 Deleting a workspace

To delete a workspace, an SICE shall send a WorkspaceDeletePDU to all SICEs in the session in the manner indicated in Table 6-3. The content of the WorkspaceDeletePDU is shown in Table 8-12. For a synchronized workspace, it shall send this SIPDU three separate times on each of the three priority channels: high, medium and low. This is done to allow a terminal which might wish to save the contents of a deleted workspace locally, to do so without ambiguity ensuring that all data bound for that workspace prior to the deletion has been consistently received at all nodes.

Table 8-12 – WorkspaceDeletePDU

Parameter	Description
workspaceIdentifier	This specifies the workspace being deleted. This may be either an active or archived workspace, although this shall not be an archived workspace if the Archive-Supported capability is not present in the negotiated capability set. In the case of an active workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace. In the case of an archived workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace except the modificationTime parameter shall correspond to the current time rather than the time of creation.
reason	Reason for workspace deletion. May be either userInitiated, insufficientStorage or a nonStandardReason.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Any node that receives a WorkspaceDeletePDU on any of the three priority channels shall first check the workspace handle to determine if the receiving SICE has a copy of this workspace. If so, it may either delete the workspace immediately, or, if it wishes to maintain a local copy of the deleted workspace in a form consistent with that which might be maintained on other workspaces, it may continue to apply information bound to this workspace from each of the priority channels up until the time that it receives the WorkspaceDeletePDU on that channel. Once it has received all three WorkspaceDeletePDUs, it has the workspace in its final form and may maintain it for local use. Note that this procedure only ensures consistency of the final form of the workspace if the workspace had been designated as synchronized.

8.4.4 Editing workspace, workspace plane and view attributes

At any time, an SICE may modify the workspace, plane and/or view attributes of a workspace by sending a WorkspaceEditPDU to all nodes in the session in the manner indicated in Table 6-3. The content of the WorkspaceEditPDU is shown in Table 8-13. For a synchronized workspace, it shall send this SIPDU redundantly on all priority channels used for SIPDUs (high, medium and low). In this case, on each priority channel, data received after reception of the WorkspaceEditPDU assumes the constraints of the new attribute set. Based upon the nature of the attribute changes, the application to the workspace of data received after the WorkspaceEditPDU on a given priority channel may need to be delayed until reception of the remaining copies of the WorkspaceEditPDU on all other priority channels.

On receipt of a WorkspaceEditPDU, an SICE shall examine the workspace handle and determine whether it has a copy of this workspace. If so, the SICE shall apply the new workspace, plane and view attributes indicated in the received WorkspaceEditPDU to the local copy of the workspace.

If more than one workspace is being edited or created at the same time by different SICEs, it is possible that the three WorkspaceEditPDUs or workspaceCreatePDUs from one SICE will be interleaved with the three from the other SICEs (i.e., the order that they are received on each of the three priority channels may be different). If multiple WorkspaceEditPDUs or WorkspaceCreatePDUs specify views with focus set as the viewState, assertion of focus is applied in the order that the WorkspaceEditPDUs and WorkspaceCreatePDUs are received on the high priority channel. Similarly, if WorkspaceEditPDUs modifying workspace, plane or view attributes for the same workspace are received interleaved on the three priority channels, attribute changes are applied in the order that the WorkspaceEditPDUs are received on the high priority channel.

If the plane protection attribute is among those in the list of attributes to be modified, the receiving SICE shall check that the user ID of the source of the WorkspaceEditPDU is the same as that of any SICE listed in the protectedPlaneAccessList included in the WorkspaceCreatePDU used to create the workspace. If so, this attribute shall be modified as indicated in the PDU. If not, this attribute shall not be modified.

Table 8-13 – WorkspaceEditPDU

Parameter	Description
workspaceIdentifier	This specifies the workspace being edited. This may be either an active or archived workspace, although this shall not be an archived workspace if the Archive-Supported capability is not present in the negotiated capability set. In the case of an active workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace. In the case of an archived workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace except the modificationTime parameter shall correspond to the current time rather than the time of creation.
attributeEdits (optional)	Sequence of workspace attributes to be modified. Workspace attributes are described in Table 8-4.
planeEdits (optional)	Sequence of planes for which plane attributes are to be modified. For each plane, the plane edit parameters are described in Table 8-14.
viewEdits (optional)	List of views to create, edit or delete. For each view, the view edit parameters are described in Table 8-15.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Table 8-14 – Workspace plane edits

Parameter	Description
plane	This parameter is the number of the plane whose attributes are to be edited.
planeAttributes	Set of plane attributes to be modified. Plane attribute parameters are described in Table 8-6.

Table 8-15 – Workspace view edits

Parameter	Description
viewHandle	Unique handle indicating the view to be modified.
action	This parameter indicates the action to be taken on the view. Possible actions are createNewView, editView, deleteView and nonStandardAction. If createNewView is indicated, a new view shall be created and associated with the workspace. An optional list of view attributes may be included to describe the characteristics of the new view. If editView is indicated, the characteristics of an existing view shall be modified. A list of view attributes indicates which attributes are to be changed. If deleteView is indicated, the indicated view shall be deleted. There are no additional parameters in this case. The nonStandardAction is only allowed if a corresponding non-standard capability has been negotiated. View attribute parameters are described in Table 8-8.

8.4.5 Copying workspace contents

If an SICE wishes to copy a rectangular region of one plane of a workspace to another rectangular region of the same plane, a different plane in the same workspace, or a different workspace, it may do so by sending a WorkspacePlaneCopyPDU to all peer SICEs in the manner indicated in Table 6-3. The content of the WorkspacePlaneCopyPDU is shown in Table 8-16.

Table 8-16 – WorkspacePlaneCopyPDU

Parameter	Description
sourceWorkspaceIdentifier	This specifies the source workspace from which a portion of a plane is to be copied. This may be either an active or archived workspace, although this shall not be an archived workspace if the Archive-Supported capability is not present in the negotiated capability set. In the case of an active workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace. In the case of an archived workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace except the modificationTime parameter shall not be included.
sourcePlane	This parameter indicates the plane from which the source rectangle is to be copied. Its value may span the range (0..N – 1) where N is the number of planes in the source workspace.

Table 8-16 – WorkspacePlaneCopyPDU

Parameter	Description
destinationWorkspaceIdentifier	This specifies the destination workspace to which a portion of a plane is to be copied. This may be either an active or archived workspace, although this shall not be an archived workspace if the Archive-Supported capability is not present in the negotiated capability set. In the case of an active workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace. In the case of an archived workspace, this shall equal the value of the workspaceIdentifier in the WorkspaceCreatePDU of the corresponding workspace except the modificationTime parameter shall correspond to the current time rather than the time of creation.
destinationPlane	This parameter indicates the plane to which the source rectangle is to be copied. Its value may span the range (0..N – 1) where N is the number of planes in the destination workspace. The destination plane shall have the same editability flag setting and the same usage designator setting as the source plane.
copyDescriptor	This parameter describes the source and destination data within the designated source and destination planes. If the source and destination planes are permanent, this parameter indicates a rectangular region in the source plan to be copied to the destination (see Table 8-17). If the source and destination planes are editable, this parameter indicates a list of objects in the source plane to be copied and a corresponding set of new object handles to be used for the newly created objects in the destination plane (see Table 8-18). Note that in the case of editable planes, scaling of the copied objects as part of the copy operation is not possible. One method of accomplishing the equivalent function is to follow the WorkspacePlaneCopy operation with object edits to adjust parameters controlling object size and position.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Table 8-17 – Permanent plane copy descriptor

Parameter	Description
sourceRegion	This parameter is a pair of points which define the rectangular region of the source plane from which information is to be copied. Each component of this parameter may span the range (–21845..43690). The source information is the set of pixel values within the source rectangle.
destinationRegion	This parameter is a destination rectangle used to specify the location to which the region is copied in the destination plane. If the Soft-Copy-Scaling capability is not present in the negotiated capability set, the size of this rectangle shall exactly equal the size of the source rectangle. Otherwise, the source information is scaled by the ratio of the destination dimensions to the source dimensions prior to placing the information in the destination workspace. All of the pixels in the destination region are overwritten by values from the source region.

Table 8-18 – Editable plane copy descriptor

Parameter	Description
objectList	This parameter indicates a list of objects in the source plane to be copied and a corresponding set of new object handles to be used for the newly created objects in the destination plane. The objects from the source copy list are copied into the destination plane with a Z-order above any existing objects. The copied objects maintain the same relative Z-order relationship to each other as they did in the source plane.
destinationOffset (optional)	This parameter defines an offset to be added to the coordinates of all of the copied objects. If not present, a zero offset is assumed.
planeClearFlag	When FALSE, the destination objects are appended to the existing set of objects in the destination plane. When TRUE, all existing objects in the destination plane are deleted prior to the copy operation.

The WorkspacePlaneCopyPDU is allowed from all SICEs if the specified destination plane is designated unprotected. If the designated plane is protected, the WorkspacePlaneCopyPDU may only be transmitted by the SICEs listed in the protectedPlaneAccessList of the workspace. If an SICE receives a WorkspacePlaneCopyPDU with a destination plane specified which is protected from an SICE which is not allowed to modify this plane, the SICE shall ignore the received SIPDU.

The WorkspacePlaneCopyPDU is allowed only between planes which have identical usage designators and identical editability flags (either both editable or both permanent). If an SICE receives a WorkspacePlaneCopyPDU specifying a source and destination plane which do not meet these qualifications, the SICE shall ignore the received SIPDU.

8.4.6 Workspace drawing colour palette

Colours used within a workspace for drawing or for defining the background colour of the workspace are specified as belonging to the colour space negotiated in the capability set reference by the application roster instance for the workspace. If the Soft-Copy-Annotation capability is present in the negotiated capability set, but none of the Soft-Copy-Color capabilities, the colour space is a palette of two colours. In addition to these two colours, drawing elements may be defined to be transparent.

If a broader colour space was negotiated for use, the corresponding colour spaces may be used. Specifically, if the Soft-Copy-Color-16, Soft-Copy-Color-202 or Soft-Copy-Color-True capabilities are present in the negotiated capability set, then the 16-colour palette, the 202-colour palette, or a true-colour specifier may be used, respectively. The three palettes and true colour are defined as follows:

- 2-colour: This palette is mandatory given support for Soft-Copy-Annotation. It has two entries, black and white. The entries in this palette are listed as the first two entries in Table 8-19.
- 16-colour: This palette is a super-set of the first. It may be used only if the Soft-Copy-color-16 capability is present in the negotiated capability set. The entries in this palette are listed as the first 16 entries in Table 8-19.
- 202-colour: This palette is a super-set of the first two palettes. This palette may be used only if the Soft-Copy-Color-202 capability is present in the negotiated capability set. The entries in this palette are all of the entries listed in Table 8-19.
- True colour: If the Soft-Copy-Color-True capability is present in the negotiated capability set, the colour of drawing elements may optionally be specified as RGB colour components, each with 8-bit precision.

A gamma of 1.8 is assumed for the workspace palette. Given that the colour accuracy of the data types rendered to this palette is not critical, the colour temperature and RGB primary values are left unspecified.

Table 8-19 – Workspace drawing colour palette

Index	R	G	B
0	0	0	0
1	255	255	255
2	128	0	0
3	0	128	0
4	128	128	0
5	0	0	128
6	128	0	128
7	0	128	128
8	192	192	192
9	128	128	128
10	255	0	0
11	0	255	0
12	255	255	0
13	0	0	255
14	255	0	255
15	0	255	255
16	192	220	192
17	166	202	240
18	255	251	240
19	160	160	164
20	0	0	0
21	8	8	8
22	16	16	16
23	25	25	25
24	33	33	33
25	41	41	41
26	49	49	49
27	58	58	58
28	66	66	66
29	74	74	74
30	82	82	82
31	90	90	90
32	99	99	99
33	107	107	107
34	115	115	115
35	123	123	123
36	132	132	132
37	140	140	140
38	148	148	148
39	156	156	156
40	165	165	165
41	173	173	173
42	181	181	181
43	189	189	189
44	197	197	197
45	206	206	206
46	214	214	214
47	222	222	222
48	230	230	230
49	239	239	239

Index	R	G	B
50	247	247	247
51	255	255	255
52	0	0	0
53	0	0	63
54	0	0	127
55	0	0	191
56	0	0	255
57	63	0	0
58	63	0	63
59	63	0	127
60	63	0	191
61	63	0	255
62	127	0	0
63	127	0	63
64	127	0	127
65	127	0	191
66	127	0	255
67	191	0	0
68	191	0	63
69	191	0	127
70	191	0	191
71	191	0	255
72	255	0	0
73	255	0	63
74	255	0	127
75	255	0	191
76	255	0	255
77	0	51	0
78	0	51	63
79	0	51	127
80	0	51	191
81	0	51	255
82	63	51	0
83	63	51	63
84	63	51	127
85	63	51	191
86	63	51	255
87	127	51	0
88	127	51	63
89	127	51	127
90	127	51	191
91	127	51	255
92	191	51	0
93	191	51	63
94	191	51	127
95	191	51	191
96	191	51	255
97	255	51	0
98	255	51	63
99	255	51	127

Index	R	G	B
100	255	51	191
101	255	51	255
102	0	102	0
103	0	102	63
104	0	102	127
105	0	102	191
106	0	102	255
107	63	102	0
108	63	102	63
109	63	102	127
110	63	102	191
111	63	102	255
112	127	102	0
113	127	102	63
114	127	102	127
115	127	102	191
116	127	102	255
117	191	102	0
118	191	102	63
119	191	102	127
120	191	102	191
121	191	102	255
122	255	102	0
123	255	102	63
124	255	102	127
125	255	102	191
126	255	102	255
127	0	153	0
128	0	153	63
129	0	153	127
130	0	153	191
131	0	153	255
132	63	153	0
133	63	153	63
134	63	153	127
135	63	153	191
136	63	153	255
137	127	153	0
138	127	153	63
139	127	153	127
140	127	153	191
141	127	153	255
142	191	153	0
143	191	153	63
144	191	153	127
145	191	153	191
146	191	153	255
147	255	153	0
148	255	153	63
149	255	153	127
150	255	153	191
151	255	153	255
152	0	204	0
153	0	204	63
154	0	204	127
155	0	204	191
156	0	204	255
157	63	204	0
158	63	204	63
159	63	204	127
160	63	204	191
161	63	204	255
162	127	204	0
163	127	204	63
164	127	204	127
165	127	204	191
166	127	204	255
167	191	204	0
168	191	204	63
169	191	204	127
170	191	204	191
171	191	204	255
172	255	204	0
173	255	204	63
174	255	204	127
175	255	204	191
176	255	204	255
177	0	255	0
178	0	255	63
179	0	255	127
180	0	255	191
181	0	255	255
182	63	255	0
183	63	255	63
184	63	255	127
185	63	255	191
186	63	255	255
187	127	255	0
188	127	255	63
189	127	255	127
190	127	255	191
191	127	255	255
192	191	255	0
193	191	255	63
194	191	255	127
195	191	255	191
196	191	255	255
197	255	255	0
198	255	255	63
199	255	255	127
200	255	255	191
201	255	255	255

8.4.7 Workspace refreshing for late arrivers

SICEs may choose to implement facilities to retransmit active (non-archived) workspaces which have been deleted in response to the reception of a GCC-Application-Roster-Report indication from the GCC provider indicating that one or more new SICEs have joined the session. The retransmitted data must conform to the constraints imposed by the new capabilities list contained within the GCC-Application-Roster-Report indication. SICEs that are not late arrivers to a particular session instance, as determined by their presence in an application roster with an earlier appRosterInstanceNumber, may already have a copy of the workspace being refreshed. Because the refreshed workspace contents may not exactly match those sourced by the original workspace creator, these SICEs shall locally recreate the workspace and its contents from the sequence of refresh exchanges. Reasons for this include (but are not limited to) the fact that capabilities can change across application roster instances forcing the refresher to transcode the contents of the workspace. Also note that new data can be directed to a workspace while it is in the process of being refreshed.

To guarantee that only one SICE in a session performs this function (should multiple SICEs be capable of it), an SICE must attempt to establish itself as the refresher by first grabbing the SI-WORKSPACE-REFRESH-TOKEN. Upon successful acquisition, the SICE shall then broadcast a WorkspaceRefreshStatusPDU (see Table 8-20) with the parameters set to indicate that this SICE is the designated refresher. This indicates to other SICEs that the refresh token has been grabbed. On receipt of a GCC-Application-Roster-Report indication which indicates that a new peer SICE has joined the session, the refresher shall rebroadcast the WorkspaceRefreshStatusPDU (see Table 8-20). If the designated refresher has abruptly left the session, other SICEs will receive a GCC-Application-Roster-Report indication in which the designated refresher is no longer included. If this should occur, other SICEs may consider this as an indication that the SI-WORKSPACE-REFRESH-TOKEN is no longer grabbed and may then attempt to become the refresher using the process described above.

Should the capabilities change in such a way that the refresher cannot fulfil its obligations due to local transcoding constraints, the refresher could choose to relinquish its role so it can be assumed by a more capable session participant. If the refresher chooses to remove itself from this role, it shall first release the SI-WORKSPACE-REFRESH-TOKEN and then broadcast a WorkspaceRefreshStatusPDU (see Table 8-20) with the parameters set to indicate that this SICE is no longer the designated refresher. When other SICEs which are capable of becoming the refresher receive this PDU, they may attempt to become the session refresher in the manner described above.

An SICE functioning as the session refresher shall refresh a workspace by issuing a workspace create exchange, as described in clause 8.4.2, for that workspace followed by all data exchanges required to recreate as close an approximation as possible to the workspace's contents in the previous workspace set corresponding to the previous application roster instance. The redundant WorkspaceCreatePDUs are still to be issued on the high, medium and low priority channels as in a normal workspace create exchange for synchronized workspaces or singularly on the high priority channel in the case of unsynchronized workspaces as described in clause 8.4.2. The refresh parameter of the WorkspaceCreatePDU shall be set to TRUE, the workspaceIdentifier parameter to the same value as the workspace being refreshed and the appRosterInstance parameter to the current application roster instance number. A refresher must reuse workspace handles and optionally reuse object handles that correlate in a one-to-one fashion with those that are being refreshed. Such reuse is used to convey the association between the old and new version of the same workspace or object to SICEs in the session that were present before the late joiner that triggered the refresh events arrived.

Should an SICE, which is not the session refresher, wish to reference a workspace that has not yet been refreshed by the refresher, it may do so by issuing a workspace create exchange in the same manner that a refresher would with all the same rules and restrictions in force. An SICE shall use

this mechanism in cases where it wishes to reference a workspace that has not been refreshed yet by the refresher (this is often the case should an SICE wish to randomly access previously viewed workspaces). The use of the refresh parameter to resolve protocol race conditions in the case where multiple SICEs simultaneously attempt the operation described above follows.

If multiple SICEs attempt to simultaneously refresh a workspace, all SICEs will receive multiple WorkspaceCreatePDUs with the refresh parameter set to TRUE. All will have the same workspaceIdentifier for the workspace and all will have the same appRosterInstance parameter value. All SICEs in the session receiving a WorkspaceCreatePDU of this type shall honour the first one that is received on the high priority channel and ignore all others that reference the same workspace handle and the same appRosterInstance parameter value. All SICEs (including the refresher) that issue a WorkspaceCreatePDU to refresh a workspace, for either synchronized or unsynchronized operation, must wait to receive their own WorkspaceCreatePDU back on the high priority channel ahead of all others attempting to refresh the same workspace before continuing. In the case of unsynchronized workspaces, the WorkspaceAcknowledgePDU shall only be issued to the SICE whose WorkspaceCreatePDU was the first received on the high priority channel. In addition, the refresher must wait to complete the unsynchronized workspace create exchange as described in clause 8.4.2 before additional exchanges can be directed towards the workspace being refreshed. For synchronized workspaces, a refreshing SICE can begin to submit additional exchanges to the workspace after receiving its own WorkspaceCreatePDU back on the high priority channel (ahead of all other WorkspaceCreatePDUs attempting to refresh the same workspace for the same roster instance). Any SICE (including the session refresher) detecting that its WorkspaceCreatePDU sourced to refresh a specific workspace was preceded by one from a different SICE attempting to perform the same refresh shall not submit any refresh data to that workspace for the remainder of that application roster instance. This rule guarantees that there is only one SICE serving as the refresher for each workspace associated with an application roster instance.

If the session refresher abruptly leaves or relinquishes its role, the new refresher for the session should restart the process for workspaces that have not yet been refreshed or not had their refreshes completed. Should an SICE, which is not the session refresher, wish to reference a workspace that may not have had its refresh complete and is not being refreshed by the refresher, it should also repeat the process. Care must be taken to handle the case whereby the refresher for a given workspace abruptly leaves the conference before completing a refresh in progress.

Care should be taken when executing refresh exchanges to minimize their interference with foreground workspace traffic. SICEs refreshing a workspace should monitor the session for activity directed towards the workspace with a view set to the focus state. If activity is detected, a local mechanism should be implemented to throttle refresh traffic to non-focus workspaces to a level that is appropriate for the throughput conditions in the session. If the workspace currently being refreshed has one of its views set to the focus state, the refresher should attempt to complete the refresh operation as quickly as possible.

Other clauses within this Recommendation indicate that workspaces shall be locally treated as deleted from the session upon the first workspace create exchange received for a specific application roster instance (only if there are new session participants). For sessions where a refresher is present, the exchange that triggers this event can be one sourced by the refresher as it attempts to refresh the last workspace with a focus view in the previous application roster instance. This is one method that could be used to avoid the condition whereby new participants have to wait an arbitrary amount of time to obtain session data.

In conducted mode sessions, the session refresher need not be granted SI nor GCC conductor privileges to perform operations related to refreshing. An SICE that is not the refresher, but that wishes to refresh a workspace, shall first successfully acquire the workspacePrivilege from the conference conductor.

Table 8-20 – WorkspaceRefreshStatusPDU

Parameter	Description
refreshStatus	A flag which, if TRUE, indicates that the SICE sourcing this PDU is functioning as the session-wide refresher. If FALSE, this indicates that the SICE sourcing this PDU has ceased to function as the session-wide refresher.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.4.8 The effect of changes to the application roster

If an SICE receives a GCC-Application-Roster-Report indication from the GCC provider, it shall examine the new application roster. If the application roster indicates that no new SICEs have enrolled since the last roster instance, the SICE shall examine the received application capabilities list, and apply the rules indicated in Table 8-1 to generate the newly negotiated capability set. If either the capability set has not changed, or the capability set has been expanded (i.e., new capabilities have been added to the negotiated list, MIN capabilities have increased in their negotiated value, or MAX capabilities have decreased in their negotiated value, but the opposite has not occurred for any capability), current workspaces shall have their roster instances reset to the one reported by the new GCC-Application-Roster-Report indication and shall be bound from then on by the new capabilities set reported.

If, on the other hand, one or more new SICEs have enrolled in the session, or, due to a node re-enrolling, the negotiated capability set has been contracted (i.e., capabilities have been removed, MIN capabilities have decreased their negotiated value, or MAX capabilities have increased their negotiated value), when the next new workspace is created by any peer SICE, all existing workspaces are automatically deleted. The newly created workspace shall make use of the new capability set, and the application roster instance parameter shall be set to the new instance value indicated in the GCC-Application-Roster-Report indication. If there is an SICE which has indicated itself to be a workspace refresher, that SICE may retransmit some or all of the previously held workspaces using the new capability set (see clause 8.4.7).

If more than one GCC-Application-Roster-Report indication is received prior to creation of a new workspace, these rules apply cumulatively until a new workspace is created. If the condition arose on any of these indications which would have resulted in deletion of the existing set of workspaces, when a new workspace is finally created, the existing workspaces are deleted even if this was not the result of the last GCC-Application-Roster-Report indication received. In this case, the application roster instance parameter shall be set to the new instance value indicated in the most recently received GCC-Application-Roster-Report indication.

When a new GCC-Application-Roster-Report is received which contains new SICEs in the session, after the next creation of a workspace, all workspaces that have application roster instance numbers less than the application roster instance number in the GCC-Application-Roster-Report are automatically deleted. A workspace refresher may recreate these workspaces once deleted (see clause 8.4.7).

8.4.9 Workspace caching

When an active (non-archived) workspace is created or an existing workspace is updated with new graphical information, the total storage required to contain the set of all workspaces within the session increases. At each SICE, this increase may or may not exceed its total storage capacity. If local storage capacity is exceeded, an SICE shall follow a strict set of rules to determine which previously existing workspaces to delete. This policy imposes a consistent resource management

policy and avoids unnecessary workspace deletions in situations where multiple SICEs are simultaneously attempting to free resources.

For the purposes of implementing this policy, all SICEs shall maintain cache state information (locally) for all active (non-archived) workspaces in a session. The valid cache states for an SI workspace are described in Table 8-21.

Table 8-21 – Workspace cache states

Workspace cache state	Description
Previously-Viewed	A workspace in this state has no view that is currently the focus view but did have a focus view in the past. It also must have its preserve attribute set to FALSE.
Not-Viewed	A workspace in this state has no view that is currently the focus and has never had a focus view. In addition, its preserve attribute must be set to FALSE.
Preserved	A workspace in this state has no view that is currently the focus view and has its preserve attribute set to TRUE.
Focus	A workspace in this state has a focus view.

When a new active (non-archived) workspace is created, the following state transition rules shall be locally applied to determine the new state of both the current focus workspace (the workspace with a view in the focus state at the time the new workspace create was received) and the newly created workspace. Even if the creation of the new workspace causes local resource limits to be exceeded, the state transition rules shall be applied immediately so that the correct workspaces will be deleted as the SICE attempts to reclaim resources. A workspace's cache state must also be updated in accordance with Table 8-21 if either the preserve attribute changes or the focus status of one of the workspace's views is changed via a WorkspaceEditPDU exchange.

Table 8-22 – Workspace cache transition rules

New workspace preserve attribute	New workspace allocating a focus view	Current focus workspace preserve attribute	New workspace cache state	Current focus workspace cache state
FALSE	FALSE	FALSE	Not viewed	Focus
FALSE	FALSE	TRUE	Not viewed	Focus
FALSE	TRUE	FALSE	Focus	Previously Viewed
FALSE	TRUE	TRUE	Focus	Preserved
TRUE	FALSE	FALSE	Preserved	Focus
TRUE	FALSE	TRUE	Preserved	Focus
TRUE	TRUE	FALSE	Focus	Previously Viewed
TRUE	TRUE	TRUE	Focus	Preserved

If a local SICE's resource limits are exceeded, the SICE shall continue to delete workspaces in the following order until its resource limits are brought within acceptable limits. This is done by issuing a WorkspaceDeletePDU for each workspace to be deleted as described in clause 8.4.3. In this case, the reason flag is set to insufficientStorage:

1) Delete workspaces in the previously viewed state from least recently made focus to most recently made focus (where "made focus" is defined to be having a view set to the focus state) until resource limits are brought within acceptable limits.

   If all previously viewed workspaces are deleted and resources are not within acceptable limits:

2) Delete workspaces in the not viewed state from most recently created to least recently created until resource limits are brought within acceptable limits. The rationale for deleting workspaces in this order is that workspaces in the not viewed state will commonly be representing graphical information that has been pre-submitted in advance of use. Given that an SICE wishing to pre-submit graphical information may cause one or more peers to exceed local resource limits, the most recently to least recently created deletion rule is beneficial in that it preserves the data that is most likely to be subsequently accessed.

   If all not viewed workspaces are deleted and resources are not within acceptable limits:

3) Delete workspaces in the preserved state from most recently marked preserved to least recently marked preserved until resource limits are brought within acceptable limits.

NOTE – The previous version of this Recommendation included an editorial error specifying that "If the storage capacity is exceeded, each SICE must determine whether to either create the new workspace and delete one or more existing workspaces, or fail to create the workspace". This choice is no longer allowed in this Recommendation but it is important to expect this behaviour from nodes implementing previous versions of this Recommendation.

8.5 Bitmaps

Bitmap exchanges are used for several functions in this Recommendation. These include exchanging text (by rendering the strings locally into a bitmap), pointing, annotating with unsupported drawing elements, and exchanging photographic and document images.

The value of the destinationAddress parameter of the BitmapCreatePDU indicates the type of bitmap being created. The allowable values for many of the BitmapCreatePDU parameters differ based on the type of bitmap being created. Also note that image bitmaps have a specified checkpointing option, which is outlined in clause 8.5.1. This may be used if the creator desires synchronized incremental display of the bitmap at all SICEs during reception.

8.5.1 Creating bitmaps

Before a bitmap create exchange is initiated, an SICE shall grab the SI-BITMAP-CREATE-TOKEN if the Soft-Copy-Bitmap-No-Token-Protection capability has not been successfully negotiated. Upon completion of the exchange, the token shall be freed if it was grabbed. This token is used to prevent multiple bitmap create exchanges from happening simultaneously in a session. Note that if the destination workspace is unsynchronized, then some overlap may occur between two sequential bitmap create exchanges if MCS-SEND-DATA is used instead of MCS-UNIFORM-SEND-DATA for unsynchronized workspaces.

To initiate the bitmap create exchange within the session, the SICE shall issue a BitmapCreatePDU to all SICEs in the session by the manner specified in clause 6.3 with the parameters set according to Table 8-23.

If all of the encoded data fit into one PDU, the moreToFollow flag shall be set to FALSE signalling the completion of the exchange to the receivers, else it shall be set to TRUE and subsequent BitmapCreateContinuePDUs shall be sent to complete the broadcast of the bitmap to the session. All receivers should guarantee that they adhere to the acknowledgment style specified in the BitmapCreatePDU even if the complete exchange fits into one PDU. It is strongly recommended that applications adjust the maximum amount of data they send in one PDU to avoid latency problems.

Table 8-23 – BitmapCreatePDU parameters

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>bitmapHandle</td><td>Unique handle returned from GCC-Registry-Allocate-Handle exchange. This handle will be used to reference this bitmap in all future SIPDUs.</td></tr>
<tr><td>destinationAddress</td><td>This parameter can take several forms depending on the value of the bitmap type and if the destination is intended to be a hard-copy device or soft-copy workspace.
<table>
<tr><th>Destination type</th><th>Bitmap type</th><th>Parameter value</th><th>Sub-parameters</th></tr>
<tr><td>hard-copy</td><td>image</td><td>hardCopyDevice</td><td>NULL</td></tr>
<tr><td>hard-copy</td><td>annotation</td><td>N/A</td><td>N/A</td></tr>
<tr><td>hard-copy</td><td>pointer</td><td>N/A</td><td>N/A</td></tr>
<tr><td>soft-copy</td><td>image</td><td>softCopyImagePlane</td><td>Destination workspace handle plane ID</td></tr>
<tr><td>soft-copy</td><td>annotation</td><td>softCopyAnnotationPlane</td><td>Destination workspace handle plane ID</td></tr>
<tr><td>soft-copy</td><td>pointer</td><td>softCopyPointerPlane</td><td>Destination workspace handle</td></tr>
</table>
The ability to create a bitmap of a certain type and the constraints on the bitmap to be created depend on the negotiated capabilities corresponding to the indicated choice of this parameter value.
In the case of a soft-copy image bitmap, the destination plane must have its plane usage set to allow image information. In the case of a soft-copy annotation bitmap, the destination plane must have its plane usage set to allow annotation.</td></tr>
<tr><td>attributes (optional)</td><td>Bitmap attributes controlling certain appearance characteristics. See Table 8-24 for details.</td></tr>
<tr><td>anchorPoint (optional)</td><td>This parameter is only applicable to bitmaps positioned within workspaces (not bound for a hard-copy device). It specifies the position of the upper left corner of the displayable region of the bitmap (as specified by the bitmapRegionOfInterest) within the destination workspace. This parameter may only be used for soft-copy bitmaps (it will be ignored for hard-copy bitmaps). If this parameter is not present for a soft-copy bitmap, the anchor point is assumed to be (0,0).</td></tr>
</table>

Table 8-23 – BitmapCreatePDU parameters

Parameter	Description
bitmapSize	This parameter specifies the horizontal and vertical size of the bitmap in pixels. Note that the pixel aspect ratio of the bitmap may not be square although the workspace coordinate system assumes a square pixel reference grid. In this case, the number of pixels the bitmap spans in the workspace will be different from the number of pixels in the bitmap itself. In the case of a bitmap format which includes more than one colour component, this parameter represents the size of the largest component. The bitmapSize parameter includes a width and height sub-parameter. The allowable range of these parameters is dependent on the destinationAddress selected: Width: destinationAddress = hardCopyDevice: (1..Hard-Copy-Image-Bitmap-Max-Width) destinationAddress = softCopyImagePlane: (1..Soft-Copy-Image-Bitmap-Max-Width) destinationAddress = softCopyAnnotationPlane: (1..Soft-Copy-Annotation-Bitmap-Max-Width) destinationAddress = softCopyPointerPlane: (1..Soft-Copy-Pointing-Bitmap-Max-Width) Height: destinationAddress = hardCopyDevice: (1..Hard-Copy-Image-Bitmap-Max-Height) destinationAddress = softCopyImagePlane: (1..Soft-Copy-Image-Bitmap-Max-Height) destinationAddress = softCopyAnnotationPlane: (1..Soft-Copy-Annotation-Bitmap-Max-Height) destinationAddress = softCopyPointerPlane: (1..Soft-Copy-Pointing-Bitmap-Max-Height) The bitmapSize parameter shall reflect the actual dimensions of the bitmap encoded in the bitmapData parameter of this PDU (along with any additional BitmapCreateContinuePDUs). If a bitmap is received for which a bitmap dimension is less than that indicated by this parameter, the bitmap shall be padded to the indicated size. In the case of a hard-copy destination, it shall be padded with white; in the case of a soft-copy destination, it shall be padded with transparent. If a bitmap is received for which a bitmap dimension is greater than that indicated by this parameter, the bitmap shall be truncated to the indicated size.
bitmapRegionOfInterest (optional)	This optional parameter selects the sub-region within the associated bitmap that is to be displayed. If the destination is an editable workspace plane, an SICE is required to store the entire transmitted bitmap. The default values for the upper left and lower right offsets, if not supplied, are (0,0) for the upper left and (bitmap width –1, bitmap height –1) for the lower right. If this parameter is used, the anchorPoint and scaling parameters apply to the region of interest rather than to the full bitmap borders.

Table 8-23 – BitmapCreatePDU parameters

Parameter	Description
pixelAspectRatio	If no scaling parameter is specified, this parameter describes the pixel aspect ratio of the bitmap. Different values are allowable depending on the bitmap destination, the bitmap format and the negotiated capability set. These constraints on pixelAspectRatio are listed in Table 8-1. The range of possible values is shown in Table 8-25. When applying a video window to a workspace in the case of a non-square pixel aspect ratio, the video pixel will be mapped to one workspace pixel along its smaller dimension. Its other dimension shall be scaled according to the pixel aspect ratio. For example, if a general pixel aspect ratio of 3:2 is specified, the corresponding bitmap has pixels whose width to height ratio is 1.5. An equivalent description would be that the bitmap is comprised of pixels that are 1.5 times as wide as they are tall. In this case, the vertical dimension of each bitmap pixel shall be interpreted to correspond to a single workspace pixel (in height) when applying the bitmap to a workspace. Horizontally, one bitmap pixel shall correspond to 1.5 workspace pixels (fractional pixel handling at bitmap boundaries is a local matter left unspecified). In the case of a pixel aspect ratio of 2:3, the opposite is true where a single bitmap pixel shall be interpreted to correspond to one workspace pixel horizontally and 1.5 pixels vertically. If a scaling parameter is specified, then this parameter shall be ignored in favour of scaling the indicated region of the bitmap to fit the workspace area specified by the anchorPoint and scaling parameters.
scaling (optional)	This optional parameter is only allowed if the Soft-Copy-Scaling is part of the negotiated capability set. This parameter, if present, indicates the offset from the anchor point, in workspace coordinates, of the lower right corner of the bitmap within the workspace. This parameter may only be used for soft-copy bitmaps (it will be ignored for hard-copy bitmaps). If this parameter is not present for a soft-copy bitmap, the lower righthand corner is determined from the bitmapSize, bitmapRegionOfInterest (if present), and the pixelAspectRatio.
checkpoints (optional)	This optional parameter, when present, specifies that checkpointing is in effect for this exchange and correspondingly specifies the set of token IDs that are to be used for checkpointing. Each token ID is used to track the status of a portion of the bitmap in transit at all the receivers by the sender. If this parameter is present, receivers should immediately inhibit all the tokens in this set upon receipt of this PDU.

Table 8-23 – BitmapCreatePDU parameters

Parameter	Description
bitmapFormatHeader	Specifies the algorithm used to encode the bitmap data and associated parameters. Certain values are applicable only to certain bitmap destinations. Some of the formats listed shall not be used unless the corresponding capability has been negotiated. See Table 8-1 for required capabilities. See clause 8.5.4 for the details of each specific encoding format supported by this Recommendation. **If destinationAddress = hardCopyDevice** Choice of: 1) bitmapHeaderUncompressed; 2) bitmapHeaderT4; 3) bitmapHeaderT6; 4) bitmapHeaderT82; 5) bitmapHeaderNonStandard (only valid if the capability corresponding to the specific bitmapHeaderNonStandard is in the negotiated capability list). **If destinationAddress = softCopyImagePlane** Choice of: 1) bitmapHeaderUncompressed; 2) bitmapHeaderT81; 3) bitmapHeaderT82; 4) bitmapHeaderNonStandard (only valid if the capability corresponding to the specific bitmapHeaderNonStandard is in the negotiated capability list). **If destinationAddress = softCopyAnnotationPlane** Choice of: 1) bitmapHeaderUncompressed; 2) bitmapHeaderT82; 3) bitmapHeaderNonStandard (only valid if the capability corresponding to the specific bitmapHeaderNonStandard is in the negotiated capability list). **If destinationAddress = softCopyPointerPlane** Choice of: 1) bitmapHeaderUncompressed; 2) bitmapHeaderT82; 3) bitmapHeaderNonStandard (only valid if the capability corresponding to the specific bitmapHeaderNonStandard is in the negotiated capability list).
bitmapData (optional)	Encoded pixel data representing the bitmap (see Table 8-26).
moreToFollow	TRUE signals that more BitmapCreateContinuePDUs will follow, carrying additional data to complete the bitmap transaction. FALSE signals that the transaction is complete with this PDU and no more will follow.

Table 8-23 – BitmapCreatePDU parameters

Parameter	Description
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Table 8-24 – Bitmap attributes

Attribute	Default value	Description
viewState	unselected	This attribute is only applicable to bitmaps positioned within workspaces (not bound for a hard-copy device) that are resident in editable workspace planes or are of type pointer. It can take on one of the following values: • unselected: Bitmap should be displayed normally. • selected: Bitmap should be displayed with some unspecified visual highlighting to indicate that the bitmap is selected and that edit or delete operations to the bitmap may be imminent (not applicable to pointer bitmaps). • hidden: Bitmap should be removed from view but remain in the local database. • nonStandardViewState
zOrder	front	This attribute is only applicable to bitmaps positioned within workspaces (not bound for a hard-copy device) that are resident in an editable workspace plane. It specifies the position in the object stack within the plane that the bitmap is to initially be at. Only one object within an editable plane can be at the front, therefore the operation of setting this attribute to front for a bitmap moves the previous front object behind this one in the stacking order. Only one object within an editable plane can be at the back, therefore the operation of setting this attribute to back for a bitmap moves the previous back object in front of this one in the stacking order.
transparencyMask	all pixels non-transparent	See clause 8.5.7 for a description of this parameter. The dimensions of the transparency mask must be identical to the dimensions of the bitmap as specified in the bitmapSize parameter (even if the bitmapRegionOfInterest parameter is specified). The transparency mask shall be logically applied to the bitmap before the bitmap is applied to the workspace, i.e., the transparency mask shall be interpreted with respect to bitmap pixels and not workspace pixels. Bitmap pixels and workspace pixels may not have a one-to-one correspondence if the bitmap employs a pixel aspect ratio which is non-square. Use of this parameter is contingent upon successful negotiation of the Soft-Copy-Transparency-Mask capability.

Table 8-24 – Bitmap attributes

Attribute	Default value	Description
nonStandardAttribute	–	This attribute is specified as a nonStandardIdentifier. To be used, it must have been successfully negotiated by a corresponding non-standard capability. Its interpretation is not specified by this Recommendation. An arbitrary number of different nonStandardAttributes may be included in the attributes list.

In order to facilitate the synchronization of the display of bitmaps at all sites in a session, the transmitter of a bitmap can mandate that checkpointing be used for the exchange. This process is initiated by supplying the optional checkpoints parameter in the BitmapCreatePDU with the value set to the sequence of token IDs to be used to mark each checkpoint. All receivers shall inhibit all tokens in the sequence immediately upon receipt in the order that they appear in the sequence. Each transmitter is responsible for allocating a set of dynamic tokens for this purpose some time before their use. The number of dynamic tokens allocated is determined by the individual applications wishing to originate checkpointed bitmap exchanges and should be proportional to the product of the number of checkpoints desired per bitmap and the maximum number of simultaneously sourced bitmap create exchanges the application allows. Each token in the set shall be allocated by issuing a GCC-Registry-Assign-Token request to the local GCC provider. The parameters used in this request are as shown in Table 8-27. If the result parameter returned in the GCC-Registry-Assign-Token confirm is "successful", the token ID contained in the returned confirm primitive can be used as a checkpointing token by the application.

Table 8-25 – Pixel aspect ratios

Pixel aspect ratio	Description
square	The horizontal extent of a pixel as compared to its vertical extent is given by the ratio – 1:1.
cif	The horizontal extent of a pixel as compared to its vertical extent is given by the ratio – 12:11 (horizontal:vertical pixel ratio).
fax1	The horizontal extent of a pixel as compared to its vertical extent is given by the ratio – 385:800 (horizontal:vertical pixel size).
fax2	The horizontal extent of a pixel as compared to its vertical extent is given by the ratio – 770:800 (horizontal:vertical pixel size).
general	SICEs wishing to use pixel aspect ratios not specifically defined by this Recommendation can specify this type and its associated sub-parameters if the appropriate capability has been negotiated. The sub-parameters used to specify the pixel aspect ratio in this general case correspond to the numerator and denominator fields of the corresponding bitmap's pixel aspect ratio in horizontal:vertical ratio format. Soft-Copy-Image-Bitmap-Any-Aspect-Ratio must be successfully negotiated for soft-copy image bitmaps and Hard-Copy-Image-Bitmap-Any-Aspect-Ratio must be negotiated for hard-copy bitmap exchanges. This method of aspect ratio specification may also be used for specifying non-standard aspect ratios given a successfully negotiated non-standard capability allowing this representation.
nonStandardAspectRatio	SICEs wishing to use pixel aspect ratios not specifically defined by this Recommendation can specify a successfully negotiated non-standard capability to represent the desired pixel aspect ratio.

Table 8-26 – bitmapData parameters

Parameter	Description
dataCheckpoint (optional)	This list of token IDs can only be present if the checkpoints parameter is present in the BitmapCreatePDU and if the data parameter carries in it encoded bitmap information that, when displayed, constitutes passing one or more checkpoints as determined by the transmitter. The receiver shall uninhibit these tokens when the bitmap data carried by the corresponding data parameter is ready for display locally.
padBits (optional)	If present, it signals the receivers to ignore a specified number of bits at the end of the data field. Allowable values are (1..256).
data	This field contains the portion of encoded bitmap bitstream contained within one BitmapCreatePDU or BitmapCreateContinuePDU. The format of this data is dictated by the sub-parameters contained within the bitmapFormatHeader field of the start parameter in addition to the coding standard specification that may apply. See clause 8.5.4 for details on the format of this field for each bitmap format. The allowable size in octets is (1..8192). Note that transmitters should limit the size of this field to an appropriate value so as to minimize latency problems that can occur with large PDUs.

Table 8-27 – Parameters for GCC-Registry-Assign-Token-Request used to allocate checkpoint tokens

Parameter	Contents
ConferenceID	Provided by GCC-Application-Permission-To-Enrol indication.
RegistryKey	Registry keys are composed of the concatenation of the session key for the session with a unique ID string that is T.50-compliant. This string is formed by concatenating the one-octet string "K" with a string formed by converting a unique handle allocated using GCC-Registry-Allocate-Handle primitive to a decimal string of at least one octet in length with no leading zeros.

For bitmap create exchanges involving bitmap data streams that exceed the maximum number of octets allowable per data parameter or that are purposely broken into smaller payloads for latency minimization, multiple PDUs must be used. BitmapCreateContinuePDUs shall be issued in a manner described in clause 6.3 until all encoded pixel data has been broadcast to the session. The parameters of this PDU are described in Table 8-28. These PDUs shall be issued in order such that concatenation of their data payload at all receivers less pad bits forms an exact copy of the bitmap datastream at the transmitter. The last such PDU shall have the moreToFollow flag set to FALSE to signal the end of the transaction.

Table 8-28 – BitmapCreateContinuePDU parameters

Parameter	Description
bitmapHandle	This parameter must be specified with the same value used in the BitmapCreatePDU for this exchange.
bitmapData	Encoded pixel data representing the bitmap (see Table 8-26).
moreToFollow	A value of TRUE signals that more BitmapCreateContinuePDUs will follow carrying additional data to complete the bitmap transaction. A value of FALSE signals that the transaction is complete with this PDU and no more will follow.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

PDUs initiating the creation or editing of objects directed at a workspace plane can arrive interspersed between the set of PDUs used to create a bitmap object on the same plane. The reception of the BitmapCreatePDU (the initial PDU in the set) shall be used as the reference when determining how to apply the bitmap data relative to other object data. When the bitmap create sequence is completed, all object creation or modification PDUs received after the reception of the BitmapCreatePDU shall be rendered in the same manner as if the bitmap had been created with a single BitmapCreatePDU. Similarly, a WorkspacePlaneCopyPDU which includes this plane as the source and/or destination plane that arrives interspersed between the set of bitmap creation PDUs is treated in the same manner as if the bitmap had been created with a single BitmapCreatePDU. These rules apply whether the effected plane is permanent or editable.

If present, a checkpoint token shall be uninhibited by each receiver as its corresponding bitmapData payload is made ready for display locally. A checkpoint token (delivered in the optional checkpoint parameter of the BitmapCreatePDU and BitmapCreateContinuePDU) is considered to correspond to a bitmapData payload if it was delivered via the same PDU.

During a checkpointed transmission, the SICE sourcing the bitmap shall poll the status of the tokens specified in the checkpoints parameter of the BitmapCreatePDU at an unspecified rate by testing the checkpoint token that has been outstanding the longest and has not yet been uninhibited by all receiving SICEs. Upon determining that one or more checkpoint tokens are free, the source SICE shall issue a BitmapCheckpointPDU with the parameters set according to Table 8-29 to indicate to all receiving SICEs that the bitmap data corresponding to the tokens specified can be displayed.

Note that a race condition exists with this mechanism in that a transmitting SICE may test a checkpoint before any receiving SICEs in the session have received the BitmapCreatePDU. In this case, the checkpointing mechanism fails in that the transmitting SICE broadcasts a BitmapCheckpointPDU for the checkpoint prematurely. To minimize the probability of this occurring, the transmitting SICE can wait for some period of time before beginning to poll the first checkpoint token. A simple way of determining this time-out period, in the case of a synchronized destination workspace, is to postpone polling of a checkpoint token until the transmitting SICE receives the BitmapCreatePDU or BitmapCreateContinuePDU with which the checkpoint token was issued back via an MCS-UNIFORM-SEND-DATA indication.

Table 8-29 – BitmapCheckpointPDU parameters

Parameter	Description
bitmapHandle	This parameter references the bitmap and shall be specified with the same value used in the BitmapCreatePDU for this exchange.
passedCheckpoint	This parameter is a list of tokens corresponding to passed checkpoints by the transmitting SICE. A checkpoint is considered passed if all the receiving terminals have uninhibited it. Receiving terminals can display the portion of the bitmap in transit corresponding to the passed checkpoints specified by this parameter.
percentComplete	This parameter's value reflects the cumulative percentage of the bitmap exchange that is complete, inclusive of the bitmap information inferred by the passedCheckpoint parameter. The value range is (1..100).
nonStandardParameters (Optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Also, in the case of a synchronized destination workspace, the transmitting SICE should not inhibit or uninhibit its own checkpoint tokens when they are returned in the appropriate PDUs via an MCS-UNIFORM-SEND-DATA indication.

Should a receiving SICE wish to cause the transmitting SICE to abort the bitmap create exchange in progress, it shall issue a BitmapAbortPDU with the parameters set to values specified in Table 8-30 to the UserID channel of the transmitting SICE. Upon receipt of the BitmapAbortPDU, the transmitting SICE may choose to not issue any additional BitmapCreateContinuePDUs and broadcast a BitmapAbortPDU to all SICEs in the session via the SI-CHANNEL to signal the premature end of the exchange. Because SICEs may have begun to display the bitmap locally before the transaction was aborted, the cleanup rules outlined in Table 8-31 should be adhered to in order to maintain display consistency within the session. If the bitmap is destined for a workspace whose resource is reallocated via a subsequent workspace create, a bitmap abort request is implied and the BitmapAbortPDU shall be issued in the manner described above.

Table 8-30 – BitmapAbortPDU parameters

Parameter	Description
bitmapHandle	This parameter references the bitmap and shall be specified with the same value used in the BitmapCreatePDU for this exchange.
userID (optional)	This parameter can be optionally supplied. It is used to indicate the MCS user ID of the SICE that requested the termination of the bitmap create exchange.
reason (optional)	This parameter is set to one of unspecified, outOfPaper, noResources or a nonStandardReason. It is used to indicate the reason for the abort request to the transmitter.
message (optional)	This parameter can be optionally supplied. It is a Unicode string suitable for display to a user.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Normal termination of the bitmap create exchange is signalled by the transmitting SICE by setting the moreToFollow parameter to FALSE in either the BitmapCreatePDU, in the case of a single PDU exchange; or in the BitmapCreateContinuePDU, in the case of a multi-PDU exchange. All receiving SICEs should be sure to have uninhibited any remaining checkpoint tokens that have not already been uninhibited during the exchange.

Table 8-31 – Cleanup rules for aborting bitmap creation

Destination	Abort cleanup rule
Soft-copy editable workspace	Delete the bitmap from the display.
Soft-copy permanent workspace or hard-copy device	Commit all bitmap data received from transmitter to the display up to the last decodable full scan line or block of scan lines.

8.5.2 Deleting bitmaps

To delete bitmaps that are members of an editable plane or that are pointers, an SICE shall issue a BitmapDeletePDU whose parameters are described in Table 8-32. An SICE can only delete a pointer if it was the creator.

Table 8-32 – BitmapDeletePDU parameters

Parameter	Description
bitmapHandle	This parameter references the bitmap to be deleted.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.5.3 Editing bitmaps

To edit the attributes of bitmaps that are members of an editable plane or that are pointers, an SICE shall issue a BitmapEditPDU whose parameters are described in Table 8-33.

Table 8-33 – BitmapEditPDU parameters

Parameter	Description
bitmapHandle	This parameter references the bitmap to be edited.
attributeEdits (optional)	List of bitmap attributes and associated values to be modified. The allowable values for list members are described in Table 8-24.
anchorPointEdit (optional)	This parameter specifies the position of the upper left corner of the displayable region of the bitmap (as specified by the bitmapRegionOfInterest) within the destination workspace. If this parameter is not present, the anchor point is not modified.
bitmapRegionOfInterestEdit (optional)	This optional parameter selects the sub-region within the associated bitmap that is to be displayed. An SICE is required to store the entire transmitted bitmap if the destination plane is editable and the destination is not a hard-copy device. If this parameter is not present, the region of interest is not modified.

Table 8-33 – BitmapEditPDU parameters

Parameter	Description
scalingEdit (optional)	This optional parameter is only allowed if the Soft-Copy-Scaling is part of the negotiated capability set. This parameter, if present, indicates the offset from the anchor point, in workspace coordinates, of the lower right corner of the bitmap within the workspace. If this parameter is not present, the scaling is not modified.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.5.4 Bitmap colour definition

Depending on the bitmap format, bitmaps may be encoded using either palettized colour maps or directly mapped colour spaces.

For the directly mapped colour spaces, greyscale, RGB, CIELab or YCbCr colour spaces are supported. Depending on the bitmap format, use of each of these colour spaces requires specific capabilities to have been negotiated among all peer SICEs (see Table 8-1).

Palettized colour mapping may be used for uncompressed and JBIG encoding formats. In this case, a colour palette of the maximum size 2^P, where P is the number of bits per pixel, is transmitted as part of the bitmap header. Each entry in the palette represents a colour which is to be used when a pixel with the value equal to the index of that palette entry is transmitted. The colours within a palette may be specified using one of several possible colour spaces either RGB, CIELab or YCbCr with their optionally associated enhancement parameters. Because colours with neighbouring indices do not necessarily result in similar colours, palettized colour mapping is only sensible to use for lossless bitmap encoding formats. Optionally, one palette entry may be defined to represent the transparent colour.

A description of each of the colour spaces supported within this Recommendation is shown in Table 8-34.

Table 8-34 – Bitmap colour spaces

Colour space	Description
Greyscale	The greyscale colour space consists of single component values with the value of zero representing black and the maximum value representing white. The intermediate values vary monotonically between these two extremes. A ColorAccuracyEnhancementGreyscale parameter is optionally specifiable to allow the gamma value of the source colour space to be indicated.
RGB	The RGB colour space consists of three component values representing the intensity of the red, green and blue primaries. For each component the value of zero represents no contribution of that component and the maximum value represents the full-intensity primary colour. The intermediate values of each component vary monotonically between these two extremes. A ColorAccuracyEnhancementRGB parameter is optionally specifiable to allow the gamma value, colour temperature (in degrees Kelvin) and the red, green and blue primary values specified as CIE xy chromaticity coordinates of the source colour space to be indicated.

Table 8-34 – Bitmap colour spaces

Colour space	Description
YCbCr	The YCbCr colour space consists of three component values. The interpretation of each colour component shall by default be in accordance with [ITU-R 601-6]. A ColorAccuracyEnhancementYCbCr parameter is optionally specifiable to allow the gamma value, colour temperature (in degrees Kelvin) and the red, green and blue primary values specified as CIE xy chromaticity coordinates of the source colour space to be indicated. Alternatively, this parameter can be set to indicate an interpretation of the colour components in accordance with [ITU-R 709-5].
CIELab	The CIELab colour space consists of three component values. The interpretation of each colour component shall by default be in accordance with [ITU-T T.42]. A ColorAccuracyEnhancementCIELab parameter is optionally specifiable to indicate the colour temperature (in degrees Kelvin) and the CIELab gamut to be used. For each component, the gamut is specified as a span and offset pair. The span represents the range in L*a*b* (continuous) space corresponding to the full dynamic range in the quantized Lab space. The offset represents the value in the quantized Lab space that represents the zero value for that component in the corresponding L*a*b* space.
nonStandardColorSpace	If the appropriate non-standard capability has been negotiated among peer SICEs, a bitmap may be transmitted using a non-standard colour space. The definition of any non-standard colour spaces is beyond the scope of this Recommendation.

8.5.5 Bitmap colour component sampling ratios

For each directly mapped colour space that includes multiple image components (all except greyscale), the resolution of the pixel arrays for each of the colour components depends on the resolution mode specified. Some of the components of the bitmap may be encoded at a resolution lower than the resolution corresponding to the bitmap size indicated in the BitmapCreatePDU, depending on the resolution mode specified. A description of the resolution modes supported by this Recommendation is shown in Table 8-35. All modes are not applicable to all bitmap encoding formats.

Table 8-35 – Bitmap resolution mode

Resolution mode	Description
4:4:4	For a 4:4:4 bitmap, each colour component is maintained at the same resolution in both horizontal and vertical dimensions equal to the size indicated by the bitmapSize parameter in the BitmapCreatePDU. This resolution mode applies to greyscale, RGB, YCbCr and CIELab colour spaces.
4:2:2	This resolution mode applies to YCbCr and CIELab colour spaces. For a 4:2:2 bitmap, the luminance component is encoded at the full resolution indicated by the bitmapSize parameter in the BitmapCreatePDU. Each of the two chrominance components are encoded with half the number of pixels in the horizontal dimension. If the horizontal dimension of the luminance component is an odd number, then the horizontal dimension of the chrominance component shall be calculated by adding one to the luminance dimension and dividing the result by two. Successive chrominance pixels along a

Table 8-35 – Bitmap resolution mode

Resolution mode	Description
	scan-line represent the position corresponding to the location mid-way between successive pairs of luminance pixels. For images with odd horizontal dimensions, each bitmap format may define the method by which luminance scan lines are extended for encoding.
4:2:0	This resolution mode applies to YCbCr and CIELab colour spaces. For a 4:2:0 bitmap, the luminance component is encoded at the full resolution indicated by the bitmapSize parameter in the BitmapCreatePDU. Each of the two chrominance components are encoded with one half the number of pixels in the horizontal dimension and one half the number of pixels in the vertical dimension. If the horizontal or vertical dimensions of the luminance component are odd numbers, then the corresponding dimension of the chrominance component shall be calculated by adding one to the luminance dimension and dividing the result by two. Successive chrominance pixels along a scan-line represent the position corresponding to the location mid-way between successive blocks of 2 × 2 of luminance pixels. For images with odd horizontal or vertical dimensions, each bitmap format may define the method by which the luminance raster is extended for encoding.
nonStandardResolutionMode	If the appropriate non-standard capability has been negotiated among peer SICEs, a bitmap may be transmitted using a non-standard resolution mode. The definition of any non-standard resolution mode is beyond the scope of this Recommendation.

8.5.6 Bitmap formats

This Recommendation supports multiple bitmap coding formats, not all of which are applicable to all bitmap destinations. Each of the supported bitmap formats are described in the following clauses.

8.5.6.1 Uncompressed

A bitstream which is encoded in uncompressed format is represented as a packed one- or three-channel pixel array depending on the selected colour space and colour resolution mode. Each available colour mapping mode may or may not be allowed, depending on the negotiated capability set and the bitmap destination. See Table 8-1 for details on capability dependencies.

For an uncompressed bitmap, the bitmapFormatHeader is set to bitmapHeaderUncompressed. The content of this header is shown in Table 8-36.

The encoding of an uncompressed bitmap for each of the possible values of the colour mapping mode parameter is defined in Table 8-37. In all cases, the image pixel array is encoded from top to bottom with a left to right line scanning. No gaps are left at scan-line boundaries.

Table 8-36 – Uncompressed bitmap format header

Parameter	Description
colorMappingMode	This parameter is a choice of either direct-mapped or palettized colour mapping. **Direct-mapped** The combination of the colorSpace and resolutionMode sub-parameter values shall be limited to those allowed by the appropriate base capability corresponding to the bitmap type implied by the destinationAddress parameter of the BitmapCreatePDU (see Table 8-1). Additional colour spaces and resolution modes are valid if the corresponding capabilities have been appropriately negotiated. The appropriate optional accuracyEnhancement parameter of the ColorSpaceSpecifier may also be included to more precisely define the colour space. **Palettized** This parameter value is limited to those allowed by the appropriate base capability corresponding to the bitmap type implied by the destinationAddress parameter of the BitmapCreatePDU (see Table 8-1).This parameter value includes as sub-components a single colour palette and a specifier indicating the number of bits per pixel – either 1, 4 or 8. The maximum length of the colour palette is determined by the number of bits per pixel (up to 2, 16 or 256 entries for 1, 4 or 8 bits per pixel, respectively). The colour palette may be specified in any of the allowable colour spaces, and may include the optional accuracyEnhancement parameter. It may also include an entry to indicate one palette entry to represent transparent.

Table 8-37 – Encoding of uncompressed bitmaps

Colour mapping	Description
1-bit palettized	The 1-bit pixel array is encoded in scanning order as described above with each 8 successive bits packed into an octet. The bits are packed beginning with the most significant bit of an octet and proceeding to fill towards the least significant bit.
4-bit palettized	The 4-bit pixel array is encoded in scanning order as described above. The 4-bit pixels are encoded together (they are not encoded as separate bitplane arrays). Pairs of successive 4-bit pixel values are packed into an octet. The first 4-bit value is placed in the most significant four bits, and the second is placed in the least significant four bits.
8-bit palettized	The 8-bit pixel array is encoded in scanning order as described above. The 8-bit pixels are encoded together (they are not encoded as separate bitplane arrays). Successive encoded 8-bit pixel values are placed into successive octets.
8-bit greyscale	The 8-bit pixel array is encoded in scanning order as described above. Successive encoded 8-bit pixel values are placed into successive octets.
RGB 4:4:4	This format is encoded with the three colour components interleaved. The image is divided into locations consisting of one pixel from each of the three colour components with the same coordinate offset. For each such location, the pixels are encoded in the following order: the R component, the G component, the B component. Successive encoded 8-bit pixels are placed into successive octets. Successive locations are dealt with in scanning order.

Table 8-37 – Encoding of uncompressed bitmaps

Colour mapping	Description
YCbCr 4:4:4	This format is encoded with the three colour components interleaved. The image is divided into locations consisting of one pixel from each of the three colour components with the same coordinate offset. For each such location, the pixels are encoded in the following order: the Y component, the Cb component, the Cr component. Successive encoded 8-bit pixels are placed into successive octets. Successive locations are dealt with in scanning order.
YCbCr 4:2:2	This format is encoded with the three colour components interleaved. The image is divided into locations consisting of a horizontal pair of Y pixels and one pixel from each of the two chrominance components that are co-located within the image. For each such location, the pixels are encoded in the following order: the left-most Y pixel, the right-most Y pixel, the Cb pixel, the Cr pixel. Successive encoded 8-bit pixels are placed into successive octets. Successive locations are dealt with in scanning order. If the horizontal dimension of the luminance component is an odd number, then the right-most Y pixel in the last YYCbCr grouping in each scan line is undefined and can be set to an arbitrary value. It is recommended that this value be set to the same value as the last valid luminance pixel in the corresponding scan line. Each of these extra luminance pixels shall be discarded by the decoder.
YCbCr 4:2:0	This format is encoded with the three colour components interleaved. The image is divided into locations consisting of a 2 × 2 block of Y pixels and one pixel from each of the two chrominance components that are co-located within the image. For each such location, the pixels are encoded in the following order: the upper-left Y pixel, the upper-right Y pixel, the lower-left Y pixel, the lower-right Y pixel, the Cb pixel, the Cr pixel. Successive encoded 8-bit pixels are placed into successive octets. Successive locations are dealt with in scanning order. If the horizontal dimension of the luminance component is an odd number, then the right-most Y pixel in the last YYCbCr grouping in each scan line is undefined and can be set to an arbitrary value. It is recommended that this value be set to the same value as the last valid luminance pixel in the corresponding scan line. If the vertical dimension of the luminance component is an odd number, then the lower-left and lower-right pixels of the last row of YYYYCbCr groupings is undefined and can be set to an arbitrary value. It is recommended that these values be set to the same value as the last valid luminance pixel in the corresponding column. Each of these extra luminance pixels shall be discarded by the decoder.
CIELab 4:4:4	This format is encoded with the three colour components interleaved. The image is divided into locations consisting of one pixel from each of the three colour components with the same coordinate offset. For each such location, the pixels are encoded in the following order: the L component, the a component, the b component. Successive encoded 8-bit pixels are placed into successive octets. Successive locations are dealt with in scanning order.

Table 8-37 – Encoding of uncompressed bitmaps

Colour mapping	Description
CIELab 4:2:2	This format is encoded with the three colour components interleaved. The image is divided into locations consisting of a horizontal pair of L pixels and one pixel from each of the two chrominance components that are co-located within the image. For each such location, the pixels are encoded in the following order: the left-most L pixel, the right-most L pixel, the A pixel, the B pixel. Successive encoded 8-bit pixels are placed into successive octets. Successive locations are dealt with in scanning order. If the horizontal dimension of the luminance component is an odd number, then the right-most Y pixel in the last YYCbCr grouping in each scan line is undefined and can be set to an arbitrary value. It is recommended that this value be set to the same value as the last valid luminance pixel in the corresponding scan line. Each of these extra luminance pixels shall be discarded by the decoder.
CIELab 4:2:0	This format is encoded with the three colour components interleaved. The image is divided into locations consisting of a 2 × 2 block of L pixels and one pixel from each of the two chrominance components that are co-located within the image. For each such location, the pixels are encoded in the following order: the upper-left L pixel, the upper-right L pixel, the lower-left L pixel, the lower-right L pixel, the A pixel, the B pixel. Successive encoded 8-bit pixels are placed into successive octets. Successive locations are dealt with in scanning order. If the horizontal dimension of the luminance component is an odd number, then the right-most Y pixel in the last YYCbCr grouping in each scan line is undefined and can be set to an arbitrary value. It is recommended that this value be set to the same value as the last valid luminance pixel in the corresponding scan line. If the vertical dimension of the luminance component is an odd number, then the lower-left and lower-right pixels of the last row of YYYYCbCr groupings is undefined and can be set to an arbitrary value. It is recommended that these values be set to the same value as the last valid luminance pixel in the corresponding column. Each of these extra luminance pixels shall be discarded by the decoder.

8.5.6.2 ITU-T Recommendation T.4 (G3)

T.4 (G3) encoding of 1-bit per pixel bitmaps is a mandatory capability for terminals supporting the hard-copy capability and disallowed otherwise.

For a T.4 encoded bitmap, the bitmapFormatHeader is set to bitmapHeaderT4. The content of this header is shown in Table 8-38.

Table 8-38 – T.4 Bitmap format header

Parameter	Description
twoDimensionalEncoding	This flag, when set to TRUE indicates that the two-dimensional encoding scheme defined in [ITU-T T.4] shall be used. When set to FALSE, the one-dimensional encoding scheme defined in [ITU-T T.4] shall be used.

For T.4 encoding, only the one-dimensional coding scheme of clause 4.1 of [ITU-T T.4] and the two-dimensional coding scheme of clause 4.2 of [ITU-T T.4] are supported. Extended two-dimensional coding as well as error-limiting mode, error-correcting mode, character mode, mixed mode and file transfer mode are not supported.

The T.4 encoded bitstream is packed into the data field of the bitmapData parameter by filling in successive bits into each octet beginning with the most significant bit of each octet and filling toward the least significant bit.

8.5.6.3 ITU-T Recommendation T.6 (G4)

T.6 (G4) encoding of 1-bit per pixel bitmaps is a mandatory capability for terminals supporting the Hard-Copy-Image-Bitmap-Format-T.6 capability, and disallowed otherwise.

For a T.6 encoded bitmap, the bitmapFormatHeader is set to bitmapHeaderT6. There are no parameters to be specified within this header – all encoding parameters are contained within the T.6 encoded data.

All T.6 encoding options are allowable except for any variable length document options.

The T.6 encoded bitstream is packed into the data field of the bitmapData parameter by filling in successive bits into each octet beginning with the most significant bit of each octet and filling toward the least significant bit.

8.5.6.4 ITU-T Recommendation T.81 (JPEG)

[ITU-T T.81] (JPEG) is an image compression standard optimized for encoding continuous tone colour images at a variety of quality levels. The composition of a JPEG bitstream and the associated encoding and decoding algorithm is specified in [ITU-T T.81].

For a JPEG bitmap, the bitmapFormatHeader is set to bitmapHeaderT81. The content of this header is shown in Table 8-39. Because JPEG omits the specification of the colour space of the coded image, a colour space specifier is provided as the only parameter to the JPEG bitmap format header.

The JPEG encoded bitstream is packed into the data field of the bitmapData parameter by filling in successive bits into each octet beginning with the most significant bit of each octet and filling toward the least significant bit.

Note that SI capabilities for JPEG bitmaps couple component interleave ratios and colorspace due to their strong correlation.

See Table 8-1 (Soft-Copy-Image-Bitmap capability) for the limits on JPEG parameters.

All other JPEG modes are available as negotiable capabilities (see Table 8-1). For each negotiated capability, the allowable range of parameters which can be specified in the JPEG frame header is shown in Table 8-40.

Table 8-39 – JPEG bitmap format header

Parameter	Description
colorSpace	This parameter's value shall be limited to the those allowed by the appropriate base capability corresponding to the bitmap type implied by the destinationAddress parameter of the BitmapCreatePDU (see Table 8-1). Additional colour spaces are valid if the corresponding capabilities have been appropriately negotiated. The appropriate optional accuracyEnhancement parameter of the ColorSpaceSpecifier may also be included to more precisely define the colour space.
resolutionMode	This parameter's value shall be limited to the those allowed by the appropriate base capability corresponding to the bitmap type implied by the destinationAddress parameter of the BitmapCreatePDU (see Table 8-1). Additional resolution modes are valid if the corresponding capabilities have been appropriately negotiated.
colorPalette (optional)	This parameter is optionally provided by a transmitting SICE to suggest a palette to receiving SICEs that is suitable for rendering the JPEG bitmap to a palette-mapped display. This parameter may be ignored by receivers if present (see clause 8.5.4 for a description of colour palettes).

Allowable JPEG bitstreams for this Recommendation have the restrictions of full interchange format only (all quantizers and Huffman tables must be specified within the image bitstream).

Table 8-40 – JPEG frame header parameters

Capability	SOF_n	P	Y	X	Nf	C_i	H_i	V_i	Tq_i
Soft-Copy-Image	SOF_0	8	(1.image bitmap max. width)	(1.image bitmap max. height)	1 or 3	(0..255)	$H_0 = 1$ or $H_0 = 2$ $H_1 = 1$ $H_2 = 1$	$V_0 = 1$ or $V_0 = 1$ $V_1 = 1$ $V_2 = 1$	(0..3)
Soft-Copy-Image-Bitmap-Format-T.81-Extended-Sequential-DCT	$+SOF_1$	+12	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-Progressive-DCT	$+SOF_2$	+12	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-Spatial-DPCM	$+SOF_3$	+(2..16)	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Sequential-DCT	$+SOF_5$	=	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Progressive-DCT	$+SOF_6$	+12	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Spatial-DPCM	$+SOF_7$	=	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-Extended-Sequential-DCT-Arithmetic	$+SOF_9$	+12	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-Progressive-DCT-Arithmetic	$+SOF_{10}$	+12	=	=	=	=	=	=	=

Table 8-40 – JPEG frame header parameters

Capability	SOF_n	P	Y	X	Nf	C_i	H_i	V_i	Tq_i
Soft-Copy-Image-Bitmap-Format-T.81-Spatial-DPCM-Arithmetic	$+SOF_{11}$	+(2..16)	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Sequential-DCT-Arithmetic	$+SOF_{13}$	=	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Progressive-DCT-Arithmetic	$+SOF_{14}$	+12	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-Differential-Spatial-DPCM-Arithmetic	$+SOF_{15}$	+(2..16)	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-YCbCr-4:2:0	=	=	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-YCbCr-4:4:4	=	=	=	=	=	=	+ $H_0 = 1$ $H_1 = 1$ $H_2 = 1$	=	=
Soft-Copy-Image-Bitmap-Format-T.81-RGB-4:4:4	=	=	=	=	=	=	+ $H_0 = 1$ $H_1 = 1$ $H_2 = 1$	=	=
Soft-Copy-Image-Bitmap-Format-T.81-CIELab-4:2:0	=	=	=	=	=	=	=	+ $V_0 = 1$ $V_1 = 1$ $V_2 = 1$	=
Soft-Copy-Image-Bitmap-Format-T.81-CIELab-4:2:2	=	=	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.81-CIELab-4:4:4	=	=	=	=	=	=	+ $H_0 = 1$ $H_1 = 1$ $H_2 = 1$	=	=

= Allowable options remain unchanged over those that have been established by other negotiated capabilities or default values.

+X Add "X" to the allowable options set that has already been established by other capabilities negotiations or default value assumptions.

The abbreviations used in Table 8-40 are defined as follows:

SOF_n: Start of frame marker. The subscript n identifies the encoding process used.

P: Sample precision. Specifies the number of bits per sample of each component.

Y: Number of lines. Specifies the number of lines in the image component with the largest number of lines.

X: Number of samples per line. Specifies the number of samples per line in the image component with the largest number of samples per line.

Nf: Number of image components in the frame.

C_i: Component identifier. Assigns a label to the component in the sequence of frame component specification parameters.

H_i: Horizontal sampling factor. Specifies the relationship between the horizontal dimension of each image component.

V_i: Vertical sampling factor. Specifies the relationship between the vertical dimension of each image component.

T_{qi}: Quantization table destination selector.

For each of the colour components indicated in the JPEG frame header, the relationship between the index i and the actual colour component for each of the supported colour spaces is shown in Table 8-41.

Table 8-41 – Order of JPEG colour components

Colour space	C_0	C_1	C_2
YCbCr	Y	Cb	Cr
CIELab	L	A	B
RGB	R	G	B
Greyscale	Y	–	–

The capability Soft-Copy-Bitmap-Format-T.81-Non-Interleaved indicates the ability to support non-interleaved encoding of the colour components. Without this capability being present in the negotiated capability set, only fully interleaved encoding is allowed. In this case, the parameter Ns in a JPEG scan header shall be equal to the number of components in the frame Nf. If the Soft-Copy-Bitmap-Format-T.81-Non-Interleaved is present in the negotiated capability set, the value of Ns may be less than the value of Nf.

8.5.6.5 ITU-T Recommendation T.82 (JBIG)

[ITU-T T.82] (JBIG) is an image compression standard optimized for lossless encoding of text, half-tone and line art images. The composition of a JBIG bitstream and the associated encoding and decoding algorithm is specified in [ITU-T T.82].

Because the JBIG Recommendation does not specify the colour space of the encoded pixel data, an out-of-band bit-stream header is provided in the bitmap format header parameter of the BitmapCreatePDU. For a JBIG bitmap, the bitmapFormatHeader is set to bitmapHeaderT82. The content of this header is shown in Table 8-42. Because JBIG is lossless, it is possible to code palettized as well as greyscale pixel data. Moreover, it is possible to send the pixel data in many scanning orders. To enable the progressive display of palettized images before all bitplanes are known by the displaying entity, intermediate palettes for some or all of the intermediate bitplanes can be optionally provided by the receiver such that the display is intelligible before all bitplanes are received. These are only useful for lowest resolution layer modes of JBIG that have stripe orderings that transfer all pixel data from one bitplane before moving on to the next. This style of bitmap exchange can be used to progressively build an image in bitplane order allowing a complete raster to be initially covered faster than if all bitplanes are transmitted for each image stripe before moving to the next. See Appendix I for one possible algorithm that can be used to generate intermediate palettes for this purpose.

The JBIG encoded bitstream is packed into the data field of the bitmapData parameter by filling in successive bits into each octet beginning with the most significant bit of each octet and filling toward the least significant bit.

All parameters that control encoding options and limits for JBIG are contained within the bi-level image header (BIH) which is present at the head of all JBIG bitstreams. Allowable ranges for these parameters when JBIG is used in conjunction with this Recommendation are shown in Table 8-43. This table also specifies the additional parameters and ranges that can be negotiated.

Table 8-42 – JBIG bitmap format header

Parameter	Description
colorMappingMode	This parameter is a choice of either direct-mapped or palettized colour mapping. **Direct-mapped** The combination of the colorSpace and resolutionMode sub-parameter values shall be limited to those allowed by the appropriate base capability corresponding to the bitmap type implied by the destinationAddress parameter of the BitmapCreatePDU (See Table 8-1). Additional colour spaces and resolution modes are valid if the corresponding capabilities have been appropriately negotiated. The appropriate optional accuracyEnhancement parameter of the ColorSpaceSpecifier may also be included to more precisely define the colour space. This parameter value shall be limited to those allowed by the appropriate base capability corresponding to the bitmap type implied by the destinationAddress parameter of the BitmapCreatePDU (see Table 8-1). For the direct-mapped case, this parameter includes a ColorSpaceSpecifier used to select the bitmap colour space from the available set. The appropriate optional accuracyEnhancement sub-parameter of the ColorSpaceSpecifier may also be included to more precisely define the colour space. Direct mapped colour spaces shall have their bitplane data (per colour component) coded and transmitted in the following order: *Coding order* ($C1_{msb}$, $C2_{msb,}$ ..., CN_{msb}), ($C1_{msb-1}$, $C2_{msb-1}$, ..., CN_{msb-1}), ..., ($C1_{lsb}$, $C2_{lsb}$, ..., CN_{lsb}) where each component, CN, is an 8-bit quantity in the form (CN_{msb}, CN_{msb-1}, ..., CN_{lsb}). In the case of the RGB colour space, the red component shall be considered C1, green C2 and blue C3. A specific component ordering must accompany the definition of any new allowable standard or non-standard direct mapped colour spaces in the future.

Table 8-42 – JBIG bitmap format header

Parameter	Description
	Palette-mapped This parameters value is limited to those allowed by the appropriate base capability corresponding to the bitmap type implied by the destinationAddress parameter of the BitmapCreatePDU (see Table 8-1). This parameter includes as a sub-component a single colour palette (see clause 8.5.4) and, optionally, a selfProgressive flag indicating whether or not the bitmapPalette is suitable for the display of bitmap data as it is incrementally decoded and displayed or a series of additional colour index tables (progressivePalettes) used to form intermediate palettes for each successive bitplane by referencing colours in the colour palette. If the number of colour index tables is less than the total number of bitplanes (as indicated within the JBIG encoded bitstream), the bitmapPalette shall be used for bitplanes after the last indicated progressive palette. The length of the bitmapPalette is bounded by the total number of bitplanes. For the progressive palettes, the length shall be equal to 2^P, where P is the bitplane number (most significant bitplane = 1) for which it is intended to be used. NOTE – If an equivalent number of colour index tables as bitplanes are present, the bitmap palette is not utilized in its transmitted order. Though sending an equivalent number of colour index tables as image bitplanes is allowable, the same results can be obtained more efficiently by including no colour index table for the final bitplane so long as the bitmap palette is reordered at the transmitter such that the colour positions are one to one with respect to the colour index table that would have been used for the final bitplane.

Table 8-43 – JBIG BIH parameters

Capability	D_L	D	P	X_D	Y_D	L_0	M_x	M_y	Order byte	Options byte
Hard-Copy-Image-Bitmap-Format-T.82	0	0	(1)	(1..image bitmap max. width)	(1..image bitmap max. height)	(1.. Y_D)	(0..127)	(0..255)	HITOLO = 0 SEQ = (0,1) ILEAVE = (0,1) SMID = (0,1)	See Table 8-44
Soft-Copy-Pointing-Bitmap-Format-T.82	0	0	(1..8)	(1..image bitmap max. width)	(1..image bitmap max. height)	(1.. Y_D)	(0..127)	(0..255)	HITOLO = 0 SEQ = (0,1) ILEAVE = (0,1) SMID = (0,1)	See Table 8-44
Soft-Copy-Annotation-Bitmap-Format-T.82	0	0	(1..8)	(1..image bitmap max. width)	(1..image bitmap max. height)	(1.. Y_D)	(0..127)	(0..255)	HITOLO = 0 SEQ = (0,1) ILEAVE = (0,1) SMID = (0,1)	See Table 8-44
Soft-Copy-Image	0	0	(1..8)	(1..image bitmap max. width)	(1..image bitmap max. height)	(1.. Y_D)	(0..127)	(0..255)	HITOLO = 0 SEQ = (0,1) ILEAVE = (0,1) SMID = (0,1)	See Table 8-44
Soft-Copy-Image-Bitmap-Format-T.82-12-Bit	=	=	+12	=	=	=	=	=	=	See Table 8-44
Soft-Copy-Image-Bitmap-Format-T.82-Differential	=	(0..255)	=	=	=	($1..Y_D/2^D$)	=	=	HITOLO = (0,1) SEQ = (0,1) ILEAVE = (0,1) SMID = (0,1)	See Table 8-44
Soft-Copy-Image-Bitmap-Format-T.82-Differential-Deterministic-Prediction	=	=	=	=	=	=	=	=	=	See Table 8-44

= Allowable options remain unchanged over those that have been established by other negotiated capabilities or default values.

+X Add "X" to the allowable options set that has already been established by other capabilities negotiations or default value assumptions.

The abbreviations used in Table 8-43 are defined as follows:

D_L: Lowest resolution layer to be specified in the associated bi-level image entity (BIE).

D: Final differential layer specified.

P: Number of bitplanes.

X_D: Horizontal picture dimension (in pixels) at the highest resolution layer.

Y_D: Vertical picture dimension (in pixels) at the highest resolution layer.

L_0: Number of lines per stripe at the lowest resolution layer.

M_x: Maximum horizontal offset allowed (in pixels) for AT pixel processing.

M_y: Maximum vertical offset allowed (in pixels) for AT pixel processing.

Order byte: Set of parameters in the BIH that specifies the order in which stripe data is concatenated to form bi-level image data (BID).

Options byte: Set of options parameters.

HITOLO: Component of the order byte that indicates whether or not stripe data is sent from highest to lowest resolution layer when using resolution reduction.

SEQ: Component of the order byte that indicates whether or not corresponding stripes from all resolution layers are sent before proceeding to the next stripe, or whether all stripes within a resolution layer are sent before proceeding to the next resolution layer.

ILEAVE: Component of the order byte that indicates whether or not stripes from multiple bitplanes should be interleaved.

SMID: Component of the order byte that indicates how stripe data is interleaved.

Table 8-44 – JBIG BIH options byte

Capability	LRLTWO	VLENGTH	TPDON	TPBON	DPON	DPPRIV	DPLAST
Hard-Copy-Image-Bitmap-Format-T.82	(0,1)	0	0	(0,1)	0	0	0
Soft-Copy-Pointing-Bitmap-Format-T.82	(0,1)	0	0	(0,1)	0	0	0
Soft-Copy-Annotation-Bitmap-Format-T.82	(0,1)	0	0	(0,1)	0	0	0
Soft-Copy-Image	(0,1)	0	0	(0,1)	0	0	0
Soft-Copy-Image-Bitmap-Format-T.82-12-Bit	=	=	=	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.82-Differential	=	=	(0,1)	=	=	=	=
Soft-Copy-Image-Bitmap-Format-T.82-Differential-Deterministic-Prediction	=	=	=	=	(0,1)	(0,1)	=

The abbreviations used in Table 8-44 are defined as follows:

LRLTWO: Component of the options byte that indicates whether or not two line templates shall be used for the image.

VLENGTH: Component of the options byte that indicates whether or not the encoded images length (in scan lines) will be determined by floating marker codes in the coded bitstream.

TPDON: Component of the options byte that indicates whether or not differential layer typical prediction shall be used for the coded bitstream.

TPBON: Component of the options byte that indicates whether or not lowest resolution layer typical prediction shall be used for the coded bitstream.

DPON: Component of the options byte that indicates whether or not differential layer typical prediction shall be used for the coded bitstream.

DPPRIV: Component of the options byte that indicates whether or not a private deterministic prediction table is specified for the image.

DPLAST: Component of the options byte that indicates whether or not the last deterministic prediction table sent shall be used.

8.5.6.6 Non-standard bitmap format

Additional coding formats not in the list of supported bitmap formats are allowed if successfully negotiated. The bitmapFormatHeader, in this case, is encoded as a nonStandardIdentifier. The interpretation of this is beyond the scope of this Recommendation.

8.5.7 Transparency masks

A TransparencyMask can be optionally specified as a companion to some objects to allow transparency to be specified at the pixel level. Its format is a two-dimensional array of bits (one bit per corresponding object pixel) that indicates whether the corresponding object pixel shall be treated as transparent or not. This method of transparency control can only be used if the Soft-Copy-Transparency-Mask capability is successfully negotiated. A mask pixel value of '1' indicates that the corresponding pixel in the associated object shall be displayed. A value of '0' indicates that the corresponding object pixel shall be treated as transparent and be subject to the rendering rules established for transparent data in that plane. Table 8-45 describes the format of a TransparencyMask.

Table 8-45 – Transparency mask

Parameters	Description
bitMask	This parameter contains the bits representing the transparency mask. A choice of either uncompressed, jbigCompressed or nonStandardFormat is allowable. The format of the contents of this parameter is described below for each of the valid choices. A mask pixel value of '1' indicates that the corresponding pixel in the associated object shall be displayed. A value of '0' indicates that the corresponding object pixel shall be treated as transparent and be subject to the rendering rules established for transparent data in that plane. **uncompressed** The image pixel mask array is packed from top to bottom with a left to right line scanning. No gaps are left at scan-line boundaries. Each 8 successive bits are packed into an octet. The bits are packed beginning with the most significant bit of an octet and proceeding to fill towards the least significant bit. If the number of mask pixels is not a multiple of eight, the final octet shall be padded with zeros and the receiver shall ignore these bits. **jbigCompressed** The jbigCompressed pixel mask shall comply with the coding procedure defined in [ITU-T T.82] (single bitplane only). Resolution reduction mode is disallowed. All pixel dimension parameters in the JBIG bitstream shall match the resolution of the pixel mask exactly. The JBIG encoded bitstream shall be packed into the data field of the transparencyMask parameter by filling in successive bits into each octet beginning with the most significant bit of each octet and filling toward the least significant bit. **nonStandardFormat** nonStandardFormat indicates the use of a format specified outside this Recommendation. It is allowed only if the corresponding non-standard capabilities are successfully negotiated.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.6 Pointers

Pointers may be applied to any workspace in a session if the Soft-Copy-Pointing capability is present in the negotiated capability set. Pointers are created by creating a bitmap using the BitmapCreatePDU with the destinationAddress parameter set to softCopyPointerPlane, possibly in conjunction with a BitmapCreateContinuePDU (see clause 8.5.1). When a pointer bitmap is created, its destination is not one of the ordinary workspace planes, but a virtual plane which is interpreted to be in front of all other planes in the workspace.

Once created, the position or other attributes of a pointer bitmap may be changed by making use of the BitmapEditPDU or may be deleted using the BitmapDeletePDU as described in clauses 8.5.2 and 8.5.3. Unlike other bitmap types, a pointer bitmap may be edited or deleted without the need to have the Soft-Copy-Plane-Editing capability present in the negotiated capability list. Also unlike other bitmap types, editing or deletion of a pointer bitmap is only allowed by the SICE which created it. An SICE shall ignore any received BitmapEditPDU or BitmapDeletePDU which refers to a pointer bitmap but is not sourced from the same SICE which issued the BitmapCreatePDU for that

pointer. If an SICE receives a new application roster in which the SICE that owns a pointer is no longer present, that pointer shall be considered deleted.

As with other bitmap types, a receiving node may wish to abort a pointer bitmap. It does so by issuing a BitmapAbortPDU as described in clause 8.5.1. The response to this request is exactly the same as that of other bitmap types in the case of an editable workspace plane as also described in clause 8.5.1 (the aborted bitmap is deleted in its entirety).

With the Soft-Copy-Pointing capability in the negotiated capability set, and no other pointing capabilities, a pointing bitmap may be up to 32 by 32 pixels in size, and may be transmitted in uncompressed format. A larger bitmap size may be used only if the capabilities Soft-Copy-Pointing-Bitmap-Max-Width or Soft-Copy-Pointing-Bitmap-Max-Height are negotiated to a larger value in the capability set. JBIG encoding format may be used if the Soft-Copy-Pointing-Bitmap-Format-T.82 capability is present in the negotiated capability set. The limitations of these two encoding formats for the case of pointer bitmaps is described in clause 8.5.6.

8.7 Video windows

8.7.1 Creating video windows

To create a video window to encapsulate an out-of-band video stream on an SI workspace an SICE shall issue a VideoWindowCreatePDU in the manner specified by clause 6.3 with parameters set according to Table 8-46. This PDU shall only be directed towards workspace planes that are editable and that have their usage designator set to include "image". Creating video windows on permanent workspace planes is disallowed and considered an error. The above imply that the Soft-Copy-Image, Soft-Copy-Plane-Editing and the Soft-Copy-Video-Window capability must be successfully negotiated to utilize this exchange. The Soft-Copy-Image and Soft-Copy-Plane-Editing capabilities require unanimous decision to be considered successfully negotiated where the Soft-Copy-Video-Window capability may be considered successfully negotiated if it is advertized by more than one SICE. It is left to the discretion of the video window creating terminals implementation to further qualify the use of video windows based on unanimous participation among all SICEs. This flexibility is intended to allow terminals that are not participating in the video portion of a conference to not interfere with the operation of video windowing (if this is the desired behaviour).

This facility is used to allow the placement and management of video streams related to the same conference that the SICE is participating in to be coordinated with and contained within an SI workspace plane. No video data is transferred via the SI protocol. The mechanism simply provides a means to reference and manage the presentation of out-of-band video streams. It is assumed that terminals utilizing this function can determine the bitstream format and method of transmission of the video stream to be encapsulated via an out-of-band means.

The plane access protection mechanism can be used to limit the SICEs that are permitted to create video windows on any particular workspace plane. Any limits imposed are initially at the discretion of the workspace creator. Care should also be taken to appropriately synchronize dimension changes or changes in availability in any out-of-band video stream with the corresponding video window object.

Table 8-46 – VideoWindowCreatePDU

Parameter	Description
videoWindowHandle	Unique handle returned from GCC-Registry-Allocate-Handle exchange. This handle will be used to reference this video window in all future SIPDUs.
destinationAddress	This parameter consists of a VideoWindowDestinationAddress which has one possible value SoftCopyImagePlaneAddress which consists of a workspaceHandle identifying the handle of the destination workspace and a plane ID indicating the plane to which the drawing element shall be created.
videoSourceIdentifier	This parameter is used to specify the out-of-band video stream whose contents should be scaled to fit and rendered within the workspace region defined by the anchor point and extent parameters. The four allowable choices are described below: **default** The video stream to be rendered within the video window shall be selected by the receiving terminal. This choice should only be used under circumstances that allow receiving terminals to infer the proper video stream for rendering in the video window (e.g., point to point video communications with only one received stream). **h243SourceIdentifier** A two-octet field. The first octet should contain the H.243 MCU ID (M), and the second octet should contain the H.243 terminal ID (T). The combination of these two parameters shall be used to determine which video stream shall be rendered within the video window. Care must be taken to handle the case where the video stream specified by the h243SourceIdentifier is made unavailable to an SICE. **h245SourceIdentifier** An integer value that references a video stream as specified in [ITU-T H.245]. The current specification of this parameter allows for unambiguous video stream identification in point-to-point connections only. A mechanism to lift this restriction may be provided in future version of [ITU-T H.245]. **dSMCCConnBinder** A DSM-CC connection binder is used to reference DSM-CC video streams. DSM-CC defines a connection binder as a sequence of all taps used for communication with a given object. The DSM-CC user-to-user definition of a tap establishes a link from an upper layer object reference to a lower layer communication channel. For example, communication with an audio/video stream object may require two separate communication paths that would be linked together within one connection binder. The association tag within a tap has end-to-end significance, even when multiple networks are crossed between the communicating endpoints. How an association tag is defined to be unique within a conference when it applies to multiple nodes is outside the scope of this Recommendation. A tap also contains an identifier, a tap use, and an optional selector for application level multiplexing (see clause 5.6.1 of [ISO/IEC 13818-6], MPEG DSM-CC IS, for additional details).

Table 8-46 – VideoWindowCreatePDU

Parameter	Description
	videoIdentifier This definition of this parameter is not specified by this Recommendation. If defined by an outside Recommendation, its contents shall be used to determine the video stream that shall be rendered within the video window. **nonStandardSourceIdentifier** The definition of this parameter is not specified by this Recommendation and its use is subject to successfully negotiating corresponding non-standard capabilities.
attributes (optional)	Video window attributes controlling certain appearance characteristics. See Table 8-47 for details.
anchorPoint (optional)	This parameter specifies the position of the upper left corner of the displayable region of the video window (as specified by the videoWindowRegionOfInterest) within the destination workspace. If this parameter is not present, the anchor point is assumed to be (0,0).
videoWindowSize	This parameter specifies the horizontal and vertical size of the video window in pixels. Note that the pixel aspect ratio of the video window may not be square although the workspace coordinate system assumes a square pixel reference grid. In this case, the number of pixels the video window spans in the workspace will be different from the number of pixels in the video window itself.
videoWindowRegionOfInterest (optional)	This optional parameter selects the sub-region within the associated video window that is to be displayed. The default values for the upper left and lower right offsets, if not supplied, are (0,0) for the upper left and (video window width –1, video window height –1) for the lower right. If this parameter is used, the anchorPoint and extent parameters apply to the region of interest rather than to the full video window dimensions. The videoWindowRegionOfInterest parameters are relative to the video streams pixel coordinate system (not that of the workspace). Note that video stream encodings and pixel formats are beyond the scope of this Recommendation.
pixelAspectRatio	If no scaling parameter is specified, this parameter describes the pixel aspect ratio of the videoWindow. The only allowable values are CIF and square unless additional non-standard values have been successfully negotiated. A description of these values can be found in Table 8-25. When applying a video window to a workspace in the case of a non-square pixel aspect ratio, the video pixel will be mapped to one workspace pixel along its smaller dimension. Its other dimension shall be scaled according to the pixel aspect ratio. See Table 8-23 for an example of how this parameter is interpreted. If a scaling parameter is specified, then this parameter shall be ignored in favour of scaling the indicated region of the video stream to fit the workspace area specified by the anchorPoint and scaling parameters.

Table 8-46 – VideoWindowCreatePDU

Parameter	Description
scaling (optional)	This parameter is only allowed if the Soft-Copy-Scaling is part of the negotiated capability set. This parameter, if present, indicates the offset from the anchor point, in workspace coordinates, of the lower right corner of the video window within the workspace. If this parameter is not present for a video window, the lower right hand corner is determined from the videoWindowSize, videoWindowRegionOfInterest (if present) and the pixel-AspectRatio.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Table 8-47 – Video window attributes

Parameter	Description
transparencyMask	Used to specify which video pixels are to be considered transparent and processed according to the rules of transparency defined in this Recommendation. See clause 8.5.7 for a description of this parameter. The dimensions of the transparency mask must be identical to the dimensions of the video data stream encapsulated by the associated video window. The transparency mask shall be logically applied to the video contents on a continuing basis before each video frame update is applied to the workspace. Use of this parameter is contingent upon unanimous negotiation of the Soft-Copy-Transparency-Mask capability.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.7.2 Deleting video windows

To delete video window from an SI workspace, an SICE shall issue a VideoWindowDeletePDU in the manner specified by clause 6.3 with parameters set according to Table 8-48. Receiving SICEs shall delete the corresponding video window from the SI workspace plane it resides within upon receipt.

Table 8-48 – VideoWindowDeletePDU

Parameter	Description
videoWindowHandle	This parameter references the video window being deleted and shall be specified with the same value used in the VideoWindowCreatePDU that created the object.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.7.3 Editing video windows

To edit a video window's parameters and attributes, an SICE shall issue a VideoWindowEditPDU in the manner specified by clause 6.3 with parameters set according to Table 8-49.

Table 8-49 – VideoWindowEditPDU

Parameter	Description
videoWindowHandle	This parameter references the video window being edited and shall be specified with the same value used in the VideoWindowCreatePDU that created the object.
videoSourceIdentifierEdit (optional)	videoSourceIdentifier edit. See Table 8-46 for details.
attributesEdits (optional)	Video window attributes edits. See Table 8-47 for details.
videoWindowRegionOfInterestEdit (optional)	videoWindowRegionOfInterest edit. See Table 8-46 for details.
anchorPointEdit (optional)	anchorPoint edit. See Table 8-46 for details.
videoWindowSizeEdit (optional)	videoWindowSize edit. See Table 8-46 for details.
pixelAspectRatioEdit (optional)	pixelAspectRatio edit. See Table 8-46 for details.
scalingEdit (optional)	scaling edit. See Table 8-46 for details.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.8 Text

The definition of a text exchange protocol is left for further study. PDUs for font management and text creation, deletion and editing have been included in the overall SI PDU syntax to facilitate the extension of this Recommendation in the future.

8.9 Drawn graphical elements

Drawing information may be transmitted to any workspace plane that has the annotation flag set in its usage designator. This may only be set if the Soft-Copy-Annotation capability is included in the negotiated capability set (the number of SICEs with this capability need be greater than one for this capability to appear in the negotiated capability set).

Drawing information may be sent to either permanent or editable workspace planes. In the case of permanent planes, the drawing commands overwrite the pixel values over which the drawing objects pass. In the case of editable planes, the drawing elements are treated as separate editable objects. The attributes of these objects may, in this case, be edited as long as the workspace and object continue to exist. Drawing elements may also be deleted from an editable workspace plane.

8.9.1 Creating drawing elements

A drawing element is created by sending a DrawingCreatePDU to all peer SICEs in a session. This is done using MCS data primitives as described in Table 6-3. The parameters of the DrawingCreatePDU are shown in Table 8-50. This SIPDU shall only be sent if the Soft-Copy-Annotation capability is included in the negotiated capability set.

Table 8-50 – DrawingCreatePDU

Parameter	Description
drawingHandle (Optional)	Unique handle allocated by GCC using the GCC-Registry-Allocate-Handle primitive. This handle is used to identify this drawing element in all future references. It is needed only if the drawing element is destined for an editable workspace plane.
destinationAddress	This parameter consists of a SoftCopyDataPlaneAddress which consists of a workspaceHandle identifying the handle of the destination workspace and a plane ID indicating the plane to which the drawing element shall be created.
drawingType	This parameter indicates the shape of the drawing element, either point, openPolyline, closedPolyline, rectangle, ellipse or nonStandardDrawingType. The ellipse setting may only be used if the Soft-Copy-Annotation-Drawing-Ellipse capability is present in the negotiated capability set. The nonStandardDrawingType may only be used if the corresponding non-standard capability is present in the negotiated capability set. See clause 8.9.4 for a description of this parameter.
attributes (optional)	Drawing element attributes controlling certain appearance characteristics. See Table 8-51 for details.
anchorPoint	This parameter specifies the origin of the drawing element within the workspace. All other points, including the remaining control points and axis of rotation, are specified relative to this control point. The anchor point may be specified in the range (–21845..43690).
rotation (optional)	This parameter indicates an angle of rotation to be applied to the drawing element and the axis of rotation relative to the anchorPoint. Use of this parameter is conditional upon successful negotiation of the Soft-Copy-Annotation-Drawing-Rotation capability. The angle is specified in integer units of minutes of arc in the range (0..21599). The angle is specified as a counterclockwise rotation. The axis of rotation is a positional offset from the anchor point specified in the range (–32768..32767). If this parameter is not specified, an angle of zero degrees (no rotation) is assumed.
sampleRate (optional)	The sample rate is an optional parameter which only applies for certain drawing types. For a point, open polyline, or closed polyline, the sample rate parameter provides a recommended rate at which the sequence of control points (and corresponding connecting lines in the case of polylines) should be displayed. This may be useful for preserving the apparent drawing rate of the user at receiving sites. The value of this parameter is specified in units of samples per second. A receiving terminal may choose to ignore this parameter. For drawing types other than point, open polyline, or closed polyline, or non-standard, this parameter shall not be included in the DrawingCreatePDU. If received, in such a case, this parameter shall be ignored. The interpretation of this parameter in the case of a non-standard drawing type is beyond the scope of this Recommendation.

Table 8-50 – DrawingCreatePDU

Parameter	Description
pointList	This parameter is a list of control points if there are any needed in addition to the anchor point. The first control point is specified relative to the anchor point. Successive control points following this are encoded relative to the previous point in the point list. When received, however, the value of each control point shall be locally translated into offsets from the anchor point so that later editing of intermediate points does not effect the location of subsequent points in this list. Later editing of the anchor point, however, does directly effect the positions of the other points. The coordinates of these points are bounded to either (–8..7), (–128..127), or (–32768..32767). Points in the point list which extend beyond the (–21845..43690) range of a workspace coordinate point shall be ignored. In the case of drawing types point, open polyline, or closed polyline, there may be from 0 to 255 points in the list. In the case of a rectangle or an ellipse, there shall be exactly one point. See clause 8.9.4.1 for a description of how the control points in the point list are interpreted for each case.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Table 8-51 – Drawing attributes

Attribute	Default value	Description
penColor	black	This attribute indicates the colour to be used to draw the line portion of this drawing element (as opposed to the fill region). The pen colour may be specified in the colour space allowed given the negotiated capability set for the designated workspace (see clause 8.4.6). If the true colour capability has been negotiated, the colour value may be specified either using the palette or as a true-colour value. The pen colour may also be specified as transparent. In the case of a permanent plane, drawing using the transparent colour sets the modified pixels to transparent, erasing what was on that plane. In the case of an editable plane, the transparent colour simply makes the drawing object invisible – objects which are below it can still be seen through. If the pen colour is not specified, a value of black is used. If the Soft-Copy-Plane-Edit capability is present in the negotiated capability list and if the workspace plane has been designated editable, then the pen colour can later be changed using the DrawingEditPDU.

Table 8-51 – Drawing attributes

Attribute	Default value	Description
fillColor	no fill	The optional fill colour parameter determines whether or not a drawing element will be filled and, if so, the colour of the filled region. If the attribute is not present in the attribute list, the drawing element is not filled. If the attribute is present, it specifies the colour of the fill region using a colour valid given the negotiated capability set (the same colour representations valid for the pen colour). In the case of a permanent workspace plane, if the transparent colour is specified, the fill region in the plane is set to the transparent colour. For an editable plane, this is equivalent to choosing not to fill, in that the fill region will have no effect on the resulting image. If the attribute is present, the fill region is determined by the drawing shape. For a point type, the fill colour has no effect (the fill region is the null set). For a closed polyline type, the fill region is the set of pixels which are enclosed by the polyline. If the polyline crosses over itself one or more times, the fill region is the set of regions enclosed by each closed loop formed by the polyline. For an open polyline, the fill region is the same region that would have been enclosed by a closed polyline with the same set of control points. For a rectangle and ellipse, the fill region is the interior of the two shapes. If the Soft-Copy-Plane-Edit capability is present in the negotiated capability list and if the workspace plane has been designated editable, then the fill colour can later be changed using the DrawingEditPDU.
penThickness	3 pixels	This attribute indicates the thickness of the line portion of this drawing element in units of pixels. If drawing is supported in a session, the pen thickness range of 3 to 16 pixels must be supported. A wider range than this may be negotiated via the Soft-Copy-Annotation-Drawing-Pen-Min-Thickness and Soft-Copy-Annotation-Drawing-Pen-Max-Thickness capabilities. The minimum thickness may be negotiated as low as one pixel, and the maximum as high as 255 pixels. If this parameter is not specified, a default value of 3 shall be used. For line thicknesses greater than one pixel, the drawing element component lines and curves shall be drawn centered about the trajectory defined by the related control point list. If the Soft-Copy-Plane-Edit capability is present in the negotiated capability list and if the workspace plane has been designated editable, then the pen thickness can later be changed using the DrawingEditPDU.

Table 8-51 – Drawing attributes

Attribute	Default value	Description
penNib	round	This attribute indicates the shape of the nib used to draw the line portion of this drawing element. The nib shape may be set to either round or square. All lines are composed of a nib continuously translated along the path of the line (or curve). In the case of a line style with dots or dashes, the nib is translated along the path of the line with periodic gaps where the nib is effectively raised and lowered again on the other side of the gap. If the pen nib parameter is not present, a circular nib shall be used. The circular nib is defined to be a solid circle of diameter equal to the pen thickness. If the Soft-Copy-Annotation-Drawing-Pen-Square-Nib parameter is present in the negotiated capability set for the designated workspace, a square nib may be used instead by specifying square as the pen nib parameter. The square nib is defined to be a solid square region with the pen thickness as the length of each side. The sides of the nib are parallel or perpendicular to either the X or Y axis of the workspace coordinate system. If the Soft-Copy-Plane-Edit capability is present in the negotiated capability list and if the workspace plane has been designated editable, then the pen nib can later be changed using the DrawingEditPDU.
lineStyle	solid	This attribute indicates the line style used to draw the line portion of this drawing element. The line style is a choice of one of the following: • solid; • dashed; • dotted; • dash-dot; • dash-dot-dot; • two-tone. For a solid line, all pixels along the path of the line are drawn using the designated pen colour. For a dashed, dotted, dash-dot or dash-dot-dot line style, the line is drawn by periodically leaving gaps across which the nib is effectively raised and lowered again on the other side of the gap. The pattern of switching is determined by which of these styles was selected. If this parameter is not specified, a solid line style shall be used as the default. A two-tone line style is a solid line that is drawn using the designated pen colour for the inner 50 per cent of the line's width, and the complementary colour for the 25 per cent of the line width along each edge. A non-standard line style may also be used if the corresponding non-standard capability has been negotiated. The rendering of line styles is not specified in this Recommendation. Because of this, the graphical content of workspaces may differ between peer SICEs as a result of differences in local functionality and terminal conditions. If the Soft-Copy-Plane-Edit capability is present in the negotiated capability list, and if the workspace plane has been designated editable, then the line style can later be changed using the DrawingEditPDU.

Table 8-51 – Drawing attributes

Attribute	Default value	Description
highlight	FALSE (not highlighted)	The highlighting flag is an optional parameter which determines whether the drawing element is treated as a solid-coloured image, or if it is treated as a semi-transparent highlight. The use of highlighting is permitted only if the Soft-Copy-Annotation-Drawing-Highlight capability is present in the negotiated capability set. If the attribute is set, the highlighting effect applies to the line portion of the drawing element as well as to the fill region, if any. When the highlighting attribute is set, the effect depends somewhat on whether the drawing element is directed at a permanent or editable workspace plane. In either case, the result should appear the same, but the means to achieve this are somewhat different. In the case of an editable plane, the attributes of the drawing element are set to indicate the use of highlighted colours. When the image is rendered, instead of occluding objects and planes behind the highlighted object, the colour that would have resulted from these objects is modified in such a way that it appears that the objects and planes behind the drawing element is covered by a semi-transparent object of the designated colour. The detailed rule for making the object appear this way shall be locally defined. In the case of a permanent plane, when the drawing element is created in a plane, it modifies the content of the plane to semi-transparent rather than solid colours. The semi-transparent colours affect the planes behind the designated plane in the manner described for the editable case (also, see clause 8.4.1.1). In this case, however, the values of the semi-transparent colour at each pixel on the designated plane over which the drawing object passes are not necessarily all the same. For pixels in the designated plane that had previously been transparent, the colour of the drawing object at that pixel is used as is. For pixels that were previously set to a colour (either solid or semi-transparent), the pixel value is modified to a new semi-transparent colour that is chosen to make it appear as if the previous colour were overwritten with the semi-transparent colour specified for the drawing element. The detailed rule for choosing this colour shall be locally defined. If the Soft-Copy-Plane-Edit capability is present in the negotiated capability list, and if the workspace plane has been designated editable, then the highlight flag can later be changed using the DrawingEditPDU.

Table 8-51 – Drawing attributes

Attribute	Default value	Description
viewState	unselected	The view state is an optional parameter which only affects drawing elements on editable workspace planes. For a permanent plane, this parameter shall be ignored. This parameter may be set to either selected, unselected, hidden, or nonStandardViewState (defined below). If the parameter is not present in the attribute list, the unselected setting shall be assumed. If the attribute is set to the selected state, the form of the drawing element may be locally modified to indicate that the object is in the selected state. The method used to indicate the selected state shall be locally defined. If the attribute is set to the hidden state, the drawing element should be removed from view but remain in the local database so that it may later be restored to view. A nonStandardViewState may be specified only if the corresponding non-standard capability is part of the negotiated capability set. Allowed values for the viewState parameter: **unselected**: Drawing element should be displayed normally. **selected**: Drawing element should be displayed with some unspecified visual highlighting to indicate that the bitmap is selected and that edit or delete operations to the bitmap may be imminent (not applicable to pointer bitmaps). **hidden**: Drawing element should be removed from view but remain in the local database. **nonStandardViewState** The viewState can later be changed using the DrawingEditPDU.
zOrder	front	zOrder (drawing depth order) is an optional parameter which is used to determine the stacking order of objects within an editable plane. For a permanent plane, this parameter shall be ignored. This parameter may be set to either front or back. If set to front, the drawing element is placed in front of all other objects in the plane. If set to back, the drawing element is placed in back of all other objects on the plane. Unlike other attributes, this attribute does not necessarily persist as new drawing elements are created, or as existing drawing elements are edited. If a drawing element is placed at the front, for example, if the depth order attribute of another drawing element is set to front, the previous drawing element is no longer at the front of all objects on the plane. This parameter is allowed only if the Soft-Copy-Plane-Edit capability is present in the negotiated capability list and if the workspace plane has been designated editable. The depth order can later be changed using the DrawingEditPDU.
nonStandard Attribute	–	This attribute is specified as a nonStandardIdentifier. To be used, it must have been successfully negotiated by a corresponding non-standard capability. Its interpretation is not specified by this Recommendation. An arbitrary number of different nonStandardAttributes may be included in the attributes list.

8.9.2 Deleting drawing elements

A drawing element may be deleted by sending a DrawingDeletePDU to all peer SICEs in a session. This is done using MCS data primitives as described in Table 6-3. The parameters of the DrawingDeletePDU are shown in Table 8-52. This SIPDU shall only be sent if the Soft-Copy-Annotation and Soft-Copy-Editing capabilities are included in the negotiated capability set, and shall only be directed at a workspace plane which has been designated as editable.

Table 8-52 – DrawingDeletePDU

Parameter	Description
drawingHandle	Handle corresponding to the drawing element to be deleted. This shall be the same value as the drawingHandle specified in the DrawingCreatePDU which created this drawing element.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.9.3 Editing drawing elements

The attributes of a drawing element, or series of drawing elements, may be modified by sending a DrawingEditPDU to all peer SICEs in a session. This is done using MCS data primitives as described in Table 6-3. The parameters of the DrawingEditPDU are shown in Table 8-53. This SIPDU shall only be sent if the Soft-Copy-Annotation and Soft-Copy-Plane-Editing capabilities are included in the negotiated capability set, and shall only be directed at a workspace plane which has been designated as editable.

Table 8-53 – DrawingEditPDU

Parameter	Description
drawingHandle	Handle corresponding to the drawing element to be edited. This shall be the same value as the drawingHandle specified in the DrawingCreatePDU which created this drawing element.
attributeEdits (optional)	List of drawing attributes and associated values to be modified. The allowable values for list members are described in Table 8-51.
anchorPointEdit (optional)	This parameter specifies the position of the anchor point within the destination workspace relative to which all other control points are defined. If this parameter is not present, the anchor point is not modified.
rotationEdit (optional)	This parameter indicates an angle of rotation to be applied to the drawing element and the axis of rotation relative to the anchor point. The angle is specified in integer units of minutes of arc in the range (0..21599). The angle is specified as a counterclockwise rotation. The axis of rotation is a positional offset from the anchor point specified in the range (–32768..32767). If this parameter is not specified, the rotation angle is not modified.

Table 8-53 – DrawingEditPDU

Parameter	Description
pointListEdits (optional)	This is a parameter that may be used to alter the control points of the drawing element. If present, this specifies a list of one or more edits. Each edit consists of an initial index referencing a control point to be edited, the new value of that control point specified relative to the anchor point of the drawing element, as well as an optional sequence of control point values to be applied to those control points with successive indices following the initially specified index. The new value of each control point in this list is encoded relative to the previous one in the list, with the first being encoded relative to the value of the control point referenced by the initial index. When received, the value of each control point shall be locally translated into offsets from the anchor point so that later editing of intermediate points does not effect the location of subsequent points in this list. The additional sequence of control points, if present, are specified bounded to either (–8..7), (–128..127), or (–32768..32767). Points in the point list which extend beyond the (–21845..43690) range of a workspace coordinate points shall be ignored. In the case of drawing types point, open polyline, or closed polyline, there may be up to 255 point list edits. In these cases, if the index value corresponds to a control point already included in the drawing element, the position of that control point is modified to the new position. If the index refers to a value which was not present in the current drawing element, a new control point is added. In this way, point sequence or polylines may be appended after creation. The ability to append additional control points to an existing drawing element shall only be allowed by the SICE which created the drawing element. When appending new control points, the new indices shall be contiguously specified from the last control point index previously defined. That is, there shall be no unspecified indices between control points. If control points are defined which are not contiguous, control points with a higher number than ones which are not defined shall be ignored by the receiver. In the case of rectangles or ellipses, there shall be no more than one control point in this list.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.9.4 Drawing types

Drawing elements may be one of several basic shapes: a point (or series of points), an open polyline, a closed polyline, a rectangle or an ellipse. The characteristics of these basic shape types are described in the following clauses.

8.9.4.1 Point

A drawing element of type point is a sequence of individual points. The anchor point and the points in the point list correspond to the centres of the points to be drawn. Each point is created from the specified nib shape of the specified thickness. The control point definition is shown in Table 8-54.

Table 8-54 – Definition of point drawing shape control point list

Control point index	Description
–	Anchor point/first point to be drawn.
0	Second point to be drawn relative to anchor point.
1	Third point to be drawn relative to the previous control point.
...	
N = (0..65534)	Last point to be drawn relative to the previous control point. The DrawingCreatePDU only allows the specification of up to 255 initial control points. Subsequent DrawingEditPDU exchanges must be performed to extend the size of the control point list beyond this limit.

8.9.4.2 Open polyline

A drawing element of type open polyline is a sequence of straight lines connecting the anchor point and successive points in the point list specified in the DrawingCreatePDU. For an open polyline, there is no line connected between the anchor point and the last point in the point list. In the case of an open polyline, there shall be at least one point specified in the point list of a DrawingCreatePDU (in addition to the anchor point). If a list with less than one point is received, the drawing element shall be ignored. The control point definition is shown in Table 8-55.

Table 8-55 – Definition of open polyline drawing shape control point list

Control point index	Description
–	Anchor point.
0	First line segment endpoint relative to anchor point.
1	Second line segment endpoint relative to the previous control point.
...	
N = (1..65534)	Last line segment endpoint relative to the previous control point. The DrawingCreatePDU only allows the specification of up to 255 initial control points. Subsequent DrawingEditPDU exchanges must be performed to extend the size of the control point list beyond this limit.

8.9.4.3 Closed polyline

A drawing element of type closed polyline is a sequence of straight lines connecting the anchor point and successive points in the point list specified in the DrawingCreatePDU. For a closed polyline, a line is also connected between the anchor point and the last point in the point list. In the case of a closed polyline, there shall be at least one point specified in the point list of a DrawingCreatePDU (in addition to the anchor point). If a list with less than one point is received, the drawing element shall be ignored. The control point definition is shown in Table 8-56.

Table 8-56 – Definition of closed polyline drawing shape control point list

Control point index	Description
–	Anchor point.
0	First line segment endpoint relative to anchor point.
1	Second line segment endpoint relative to the previous control point.
...	
N = (1..65534)	Last line segment endpoint relative to the previous control point. The closing line segment is drawn to the starting point. The DrawingCreatePDU only allows the specification of up to 255 initial control points. Subsequent DrawingEditPDU exchanges must be performed to extend the size of the control point list beyond this limit.

8.9.4.4 Rectangle

A drawing element of type rectangle is a rectangular region whose upper left corner and lower right corner are specified by the anchor point and a single point in the point list, respectively. In the case of a rectangle, there shall be exactly one point specified in the point list of a DrawingCreatePDU (in addition to the anchor point). If a list with more than one point is received, the additional points shall be ignored. If a list with less than one point is received in the case of a DrawingCreatePDU, the drawing element shall be ignored. The control point definition is shown in Table 8-57.

Table 8-57 – Definition of rectangle drawing shape control point list

Control point index	Description
–	Anchor point/upper left corner.
0	Lower right corner relative to anchor point.

8.9.4.5 Ellipse

A drawing element of type ellipse is defined by a bounding rectangle whose upper left corner and lower right corner are specified by the anchor point and a single point in the point list, respectively. The size of this rectangle corresponds to the length of the two axes of the ellipse. The ellipse is positioned so that no part of it protrudes beyond the bounding rectangle. The ellipse drawing type shall not be used unless the Soft-Copy-Annotation-Drawing-Ellipse capability is present in the negotiated capability set. In the case of an ellipse, there shall be exactly one point specified in the point list of a DrawingCreatePDU (in addition to the anchor point). If a list with more than one point is received, the additional points shall be ignored. If a list with less than one point is received in the case of a DrawingCreatePDU, the drawing element shall be ignored. The control point definition is shown in Table 8-58.

NOTE – The bounding rectangle used in defining this drawing element does not correspond to an actual area of the workspace overwritten by this drawing element. Only the line which forms the ellipse itself (and optionally the fill area within the ellipse) is modified in the case of a non-editable workspace plane, or opaque in the case of an editable workspace plane.

Table 8-58 – Definition of ellipse drawing shape control point list

Control point index	Description
–	Anchor point/upper left corner of bounding rectangle.
0	Lower right corner of bounding rectangle relative to anchor point.

8.9.4.6 Non-standard

A drawing element of non-standard shape is permitted to have from 1 up to 65534 control points in the point list. The meaning of these control points is beyond the scope of this Recommendation. The control point definition is shown in Table 8-59.

NOTE – Non-standard drawing shapes should be specified such that the drawing element does not extend beyond the smallest size rectangle which can enclose all of the control points. This allows the decision of whether the rectangular copy region for the WorkspacePlaneCopyPDU surrounds the control points (to determine whether the drawing element is to be included in the copy) to be consistent with whether that copy region surrounds the actual drawing element.

Table 8-59 – Definition of non-standard drawing shape control point list

Control point index	Description
–	Anchor point/non-standard control point 0.
0	Non-standard control point 1 relative to anchor point.
1	Non-standard control point 2 relative to the previous control point.
...	
N = (0..65534)	Non-standard control point N+1 relative to the previous control point. The DrawingCreatePDU only allows the specification of up to 255 initial control points. Subsequent DrawingEditPDU exchanges must be performed to extend the size of the control point list beyond this limit.

8.10 Remote events

When a workspace is created, it may be designated to be capable of accepting remote keyboard or pointing device events from other SICEs in the session by setting the accept keyboard events and/or the RemotePointingDeviceEvents flag in the WorkspaceCreatePDU. For any workspace which has indicated either or both of these abilities, any SICE in the session may send corresponding remote events to this workspace if the creator of the workspace is still in the session.

To do this, the SICE is required to first request permission to issue the corresponding event PDUs by issuing a RemoteEventPermissionRequestPDU with the parameters set according to Table 8-60. The SICE shall then wait for a RemoteEventPermissionGrantPDU from the creator of the workspace with the destinationUserID parameter set to its MCS user ID. The event permissions list contained within this PDU defines the types of events the SICE is allowed to source without being ignored. Receipt of a RemoteEventPermissionGrantPDU at a later time with a different permission list indicates that the SICEs permissions have been changed. The RemoteEventPermissionGrantPDU is sent on the SI-CHANNEL to allow all SICEs in the session to be aware of the activity.

An SICE wishing to relinquish its ability to source a type of remote event may do so by issuing a RemoteEventPermissionGrantPDU with the RemoteEventPermissionList parameter set to the values indicating the event privileges wishing to be relinquished.

The administration of remote event privileges for a workspace is at the discretion of the workspace creator. The conductor privileges mechanism, when in conducted mode, form an additional constraint on the ability to source remote events. The source SICE must have remote event privileges granted by both the conductor and the workspace creator.

NOTE – It is recommended that workspaces created for the purpose of accepting remote events be specified to have the preserve flag set in the workspace attributes. This helps to ensure (although it does not guarantee) that the workspace will not be deleted if an SICE should create a new workspace with a view in the focus state.

Table 8-60 – RemoteEventPermissionRequestPDU

Parameter	Description
destinationAddress	This parameter shall be set to the value softCopyWorkspace with its sub-parameter set to a unique handle indicating the workspace to which the sending SICE wishes to issue remote events.
remoteEventPermissionList	Set of one or more of the following values: keyboardEvent, pointingDeviceEvent, nonStandardEvent.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Table 8-61 – RemoteEventPermissionGrantPDU

Parameter	Description
destinationAddress	This parameter shall be set to the value softCopyWorkspace with its sub-parameter set to a unique handle indicating the workspace to which the SICE indicated by the desintationUserID has been granted the permissions indicated in the remoteEventPermissionList.
destinationUserID	MCS user ID of destination node.
remoteEventPermissionList	Set of one or more of the following values: keyboardEvent, pointingDeviceEvent, nonStandardEvent. In the case of the pointingDeviceEvent permission, this entry may optionally include a handle which corresponds to the pointer bitmap which corresponds to the controlled pointing device from this SICE for this workspace.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.10.1 Remote keyboard events

If an SICE has been granted the keyboardEvent permission by the mechanism described in clause 8.10, the SICE may send a keyboard event, it shall do so by sending a RemoteKeyboardEventPDU to the SICE that created the workspace (the issuer of the WorkspaceCreatePDU). This is done in the manner indicated in Table 6-3. The content of the RemoteKeyboardEventPDU is shown in Table 8-62.

The action taken by the workspace owner on receipt of a RemoteKeyboardEventPDU is a locally defined matter beyond the scope of this Recommendation.

Table 8-62 – RemoteKeyboardEventPDU

Parameter	Description
destinationAddress	This parameter shall be set to the value softCopyWorkspace with its sub-parameter set to a unique handle indicating the workspace to which to direct the pointing device event. This shall equal the value of the workspace handle in the WorkspaceCreatePDU of the corresponding workspace.
keyModifierStates (optional)	This is a list of optional key modifiers indicating whether or not the keyCode should be conditioned by any combination modifiers. The possible modifiers are leftAlt, rightAlt, leftShift, rightShift, leftControl, rightControl, leftSpecial, rightSpecial, numberPad, scrollLock or nonStandardModifier. There may be an arbitrary number of different nonStandardModifiers included in the list. A nonStandardModifier may only be used if the corresponding non-standard event permission has been granted in the RemoteEventPermissionGrantPDU. Left and right correspond to left and right handed versions of each basic modifier. The numberPad modifier indicates whether the key code was actuated from a numeric keypad rather than a primary keypad.
keyPressState	This parameter is used to indicate the action of a key indicated by this SIPDU. It may be one of none, keyPress, keyDown, keyUp or nonStandardKeypressState. A nonStandardKeypressState may only be used if the corresponding non-standard event permission has been granted in the RemoteEventPermissionGrantPDU. The none case may be used for a key which is already in the down state if the keyModifierStates have changes. The keyPress action indicates an instantaneous key press action. This may be used, for example, to signal key-presses due to auto-repeat if the key is already in the key-down state. The keyDown and keyUp actions represent a state change of the key to a pressed or non-pressed position, respectively. Successive keyDown actions without and intervening keyUp action for the same key shall be ignored by the receiver. Successive keyUp actions without intervening keyDown actions for the same key shall be ignored by the receiver. Before relinquishing keyboard event privileges for this workspace, it is recommended that an SICE set all keys in the keyDown state to the keyUp state.
keyCode	This parameter is a choice of either a single Unicode encoded character, a choice of one of 32 function (F) keys, or a choice of one of the following edit keys: upArrow, downArrow, leftArrow, rightArrow, pageUp, pageDown, home, end, insert, delete or nonStandardKey. A nonStandardKey may only be used if the corresponding non-standard capability has been negotiated.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.10.2 Remote pointing device events

When an SICE that has been granted pointingDeviceEvent permission wishes to send a pointing device event, it shall do so by sending a RemotePointingDeviceEventPDU to the SICE which created the workspace (the issuer of the WorkspaceCreatePDU). This is done in the manner indicated in Table 6-3. The content of the RemotePointingDeviceEventPDU is shown in Table 8-63. The initial remote pointer position shall be undefined and all button states shall be assumed to be buttonUp by the SICE granting the pointingDeviceEvent permission until receipt of the first RemotePointingDeviceEventPDU. An SICE may grant the pointingDeviceEvent permission to multiple other SICEs. How multiple pointing device states are simultaneously

interpreted or merged is beyond the scope of this Recommendation. Note, though, that care must be taken to properly handle the case of an SICE with pointingDeviceEvent permission leaving the session or having its pointingDeviceEvent permission revoked with any button not in a buttonUp state.

The action taken by the workspace creator on receipt of a RemotePointingDeviceEventPDU is a locally defined matter beyond the scope of this Recommendation. Before relinquishing the pointingDeviceEvent permission for this workspace, it is recommended that the SICE send an event setting the state of all buttons to buttonUp if the last RemotePointingDeviceEventPDU sent by the SICE had any button states set to anything other than buttonUp.

If the creator of the workspace had indicated the handle of the pointer bitmap corresponding to this SICE, the SICE which is controlling that pointing device may optionally track incoming edits to the location of the pointer bitmap with this handle value. This SICE may optionally choose to alter its local display so that the corresponding pointer bitmap is positioned locally, ignoring the incoming position changes. This could be used to provide more rapid response time to movement of the local pointing device.

Table 8-63 – RemotePointingDeviceEventPDU

Parameter	Description
destinationAddress	This parameter shall be set to the value softCopyWorkspace with its sub-parameter set to a unique handle indicating the workspace to which to direct the pointing device event. This shall equal the value of the workspace handle in the WorkspaceCreatePDU of the corresponding workspace.
leftButtonState	This parameter indicates the state of the left button of the remote pointing device. Allowable values are detailed in Table 8-64.
middleButtonState	This parameter indicates the state of the middle button of the remote pointing device. Allowable values are detailed in Table 8-64.
rightButtonState	This parameter indicates the state of the right button of the remote pointing device. Allowable values are detailed in Table 8-64.
initialPoint	This parameter shall indicate the initial position within the workspace of the pointing device at the time of the event. The point is represented in workspace coordinates (see clause 8.4.1.2).
sampleRate (optional)	This is an optional parameter that indicates the rate at which the remote pointing device acquires the successive pointing device coordinates expressed in this PDU. The units are in samples/second (1..255).
pointList (optional)	This parameter is a list of differential coordinates that are specified relative to their predecessors (the first one is specified relative to the initialPoint parameter). This list is used to collapse multiple sequential pointing device translation events that have the same button states into one transaction.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.10.3 Remote printing events

At any time, an SICE may request that a workspace be printed at all nodes in a session which are capable of doing so by issuing a RemotePrintPDU to all peer SICEs in the session. This is done in the manner described in Table 6-3. The content of the RemotePrintPDU is shown in Table 8-65. In the case of an unsynchronized workspace, this SIPDU shall be transmitted on the high priority data channel. In the case of a synchronized workspace, an SICE shall send this SIPDU three separate

times on each of the three priority channels: high, medium and low. This is done to ensure that the same information is applied to the workspace before printing so that it is consistently printed at all nodes.

Table 8-64 – Pointing device button events

Button event	Description
buttonUp	Button is up.
buttonDown	Button is down.
buttonDoubleClick	Button was pressed within a locally determined time interval that constitutes a double click. If the receiving SICE has no need to distinguish double clicks then it can be interpreted as a buttonDown.
buttonTripleClick	Button was pressed within a locally determined time interval that constitutes a triple click. If the receiving SICE has no need to distinguish triple clicks then it can be interpreted as a buttonDown.
buttonQuadClick	Button was pressed within a locally determined time interval that constitutes a quad click. If the receiving SICE has no need to distinguish quad clicks then it can be interpreted as a buttonDown.
nonStandardButtonEvent	A non-standard button event has occurred. A nonStandardButtonEvent may only be used if the corresponding non-standard event permission has been granted in the RemoteEventPermissionGrantPDU.

Table 8-65 – RemotePrintPDU

Parameter	Description
destinationAddress	This parameter shall be set to the value softCopyWorkspace with its sub-parameter set to a unique handle indicating the workspace to be printed. This shall equal the value of the workspace handle in the WorkspaceCreatePDU of the corresponding workspace.
numberOfCopies (optional)	Optional indication of the number of copies desired to be printed at the remote sites. If this parameter is not present, it is assumed that one copy is to be printed.
portrait (optional)	TRUE specifies a portrait paper orientation. FALSE specifies a landscape paper orientation. If the parameter is not present, no particular orientation is preferred.
regionOfInterest (optional)	Optionally specifies a rectangular region to be printed within the designated workspace. If this parameter is not present, it is implied that the entire workspace is to be printed.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

When an SICE receives a RemotePrintPDU, if it does not have a copy of the designated workspace (if it joined the session after this workspace was created), it shall ignore this SIPDU. If the SICE is not capable of printing, it may also ignore this SIPDU.

In the case of an unsynchronized workspace, a RemotePrintPDU received on any priority channel except high shall be ignored. Upon receipt of a RemotePrintPDU on the high priority channel, an SICE which wishes to respond to print requests shall print immediately upon receiving this SIPDU.

In the case of a synchronized workspace, on receiving a RemotePrintPDU from a requesting node on any one of the three priority channels (high, medium or low), an SICE shall stop applying

updates that are received on that channel to the designated workspace. It shall continue applying workspace updates to the workspace from each of the other two priority channels until it receives a RemotePrintPDU from the same requesting node specifying the same workspace on each of these priority channels. Upon receiving each of these RemotePrintPDUs, it shall stop applying updates to the workspace received on that channel until the workspace has been printed or copied to an area to ready it for printing. While waiting for the remaining RemotePrintPDUs, if the SICE receives a GCC-Application-Roster-Report from the GCC provider that indicates that the SICE which originated the print request is no longer enrolled in the session, the SICE shall stop waiting for the remaining requests and continue applying workspace updates from all priority channels to this workspace (unless another print request is pending for the same workspace).

8.11 Archives

8.11.1 Opening archives

An archive is a collection of workspaces which may be saved beyond the extent of a single session. If an archive is present at every peer SICE in a session, it may be opened by an SICE if the Archive-Support capability is present in the negotiated capability set. Once opened, that SICE may perform workspace operations on workspaces contained within. An archive may be opened for reading, writing, creation (of a new archive), or any combination of these. Multiple SICEs may simultaneously open an archive for reading, but only one SICE may open the same archive for creation or writing. If multiple SICEs open an archive, each open shall use a unique archiveHandle. To open an archive, the ArchiveOpenPDU shall be broadcast to all peer SICEs. This is done in the manner described in Table 6-3. The content of the ArchiveOpenPDU is shown in Table 8-66. Upon receiving the ArchiveOpenPDU, all SICEs shall send an ArchiveAcknowledgePDU to the SICE requesting the open on its used ID channel. The SICE requesting the open shall wait until all peer SICEs marked as active in the current application roster have responded to the open request before proceeding with archive operations. If not all active peer SICEs acknowledge, the archive shall not be considered open. If the result parameter in any of the ArchiveAcknowledgePDUs returned indicates an unsuccessful open, the SICE issuing the ArchiveOpenPDU shall explicitly issue an ArchiveClosePDU specifying the same archive handle included in the failed open transaction. This insures that all SICEs that successfully opened the archive close it. The SICE shall also monitor any roster changes as indicated from GCC so as to be able to recognize when an active peer SICE that has not responded has left the session. If the roster changes with any new SICEs being added to the session since the roster instance that was valid at the time the archive was opened, the archive shall be closed by issuing an ArchiveClosePDU. Another ArchiveOpenPDU must be transmitted in order to perform any further archive operations.

Once an archive has been opened, it may be operated on by the SICE which opened it using any of the workspace PDUs. An SICE which has not opened an archive shall not perform any archive operations on that archive. If an archive has been opened for reading but not writing (or creation), workspaces may not be created, deleted or edited. Only workspace plane copy operations are allowed where the source is a workspace in this archive and the destination is an active workspace or a workspace in another archive. WorkspacePlaneCopy operations involving editable planes shall substitute an ordinal number for the normal GCC unique handles used to reference objects. When copying objects into an archive, the ordinal number shall be the position of the object in the EditablePlaneCopyDescriptor object list (index base 0) if the planeClearFlag is set. If the planeClearFlag is not set, the handle shall be the ordinal number representing the position of the object in the EditablePlaneCopyDescriptor object list (index base 0) added to the greatest handle value of any object that exists within the destination plane at the time of the copy. If an archive has been opened for writing (or creation) but not for reading, workspaces may be created, deleted or edited, and workspace plane copies may be performed where the source workspace is an active workspace or a workspace in another archive and the destination is a workspace in this archive. If an archive is opened for both reading and writing (or creation), any workspace operation may be

performed. When performing a workspace operation on an archived workspace, the workspace-Identifier parameter is set to the handle of the archive as indicated in the ArchiveOpenPDU, along with the entry name of the particular workspace within the archive.

Also to be included in the workspaceIdentifer for some workspace operations to archives is a modificationTime parameter. This parameter shall be included in the case that the archive is modified in any way. This includes workspace creation, editing, deletion, as well as workspace plane copies for the destination workspace. This parameter shall not be included in the workspaceIdentifier in the case of workspace creation acknowledge or ready indications, or for the source workspace in a workspace plane copy operation. When an archive is successfully modified in some way, the archive header used to identify the workspace is changed to include the new modification time as the archiveModificationTime parameter. If an archive modification fails for any reason, the modification time is not updated and the archive shall remain as it was before the failed operation was attempted. When the archive is to be opened at a later time, the header used to reference it shall include the most recent modification time as the archiveModificationTime parameter. When opening an archive, if the modification time indicated in the open request does not match that of the local archive at an SICE, the resulting result code in the ArchiveAcknowledgePDU shall include the actual header contents including the modification time. This may allow the requesting SICE to identify the differences between the available archive and the desired archive.

Table 8-66 – ArchiveOpenPDU

Parameter	Description
archiveHandle	Unique handle used to reference the archive volume once it has been opened.
mode	This parameter indicates the action to be performed on the archive. This parameter may be set to read, write, create or a combination of these. The read setting indicates that the archive shall be opened for reading. The write setting indicates that the archive shall be opened for writing. The create setting indicates that a new archive shall be created. In this case, if an archive of the same name already exists, this operation shall fail.
header	The header parameter is the identifier by which the archive shall be referenced for the duration of its existence. The header is formed from an archiveName, archiveCreationTime which indicates the creation time of the archive, and archiveModificationTime which indicates the time that the contents of the archive were last modified.
maxEntries (optional)	This parameter shall be present if the archive mode is set to create, and shall not be present otherwise. This parameter, which may span the range (1..65535) indicates the maximum number of entries which may be placed in the archive. This can be used to allow the receiving nodes to attempt to allocate space for the archive, allowing them to indicate an error condition if they do not have sufficient space.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Table 8-67 – ArchiveAcknowledgePDU

Parameter	Description
archiveHandle	Unique handle used to reference the archive volume once it has been opened.
result	Result of the archive open operation. Either archiveOpenSuccessful, archiveNotFound (when opening for reading or writing), archiveTimeIncorrect (when opening for reading or writing), archiveExists (when opening for creation), archiveOpenForWriting (when opening for writing), storageExceeded, unspecifiedError or nonStandardResult. In the case of archiveTimeIncorrect, the actual header of the archive that was found is included in the result to allow the SICE requesting the open to determine which version of the archive was present at the acknowledging SICE.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.11.2 Closing archives

When the SICE that opened an archive has completed all operations on that archive, it may be closed by broadcasting an ArchiveClosePDU to all peer SICEs. This is done in the manner described in Table 6-3. The content of the ArchiveClosePDU is shown in Table 8-68. The SICE shall send this PDU three separate times on each of the three priority channels: high, medium and low. This is done to ensure that all archive modifications made prior to closing the archive are processed. At receiving SICEs, only those operations resulting from PDUs received on a given priority channel prior to the ArchiveClosePDU on that channel are applied to the archive. This PDU is only valid if received from the SICE which opened the specified archive. From any other SICE, this PDU shall be ignored.

Table 8-68 – ArchiveClosePDU

Parameter	Description
archiveHandle	Unique handle used to reference the archive volume to be closed.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.11.3 Handling archive errors

If an error occurs in an archive as a result of an archive operation, receiving SICEs shall send an ArchiveErrorPDU to the node which originated the archive operation. This is done in the manner described in Table 6-3. The content of the ArchiveErrorPDU is shown in Table 8-69.

Table 8-69 – ArchiveErrorPDU

Parameter	Description
archiveHandle	Unique handle used to reference the archive volume to which this PDU refers.
entryName (optional)	Specifies the archive entry associated with the error if applicable.
errorCode	This parameter indicates the type of error that occurred. It is either entryNotFound (when reading, modifying or deleting an entry), entryExists (when creating an entry), storageExceeded, archiveNoLongerAvailable, unspecifiedError, or nonStandardError.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

8.12 Conducted mode operation

When a session is in conducted mode, the ability for an SICE to perform many of the operations described in clause 8 may be restricted by the conducting node. The GCC permission mechanism is used to determine whether these operations are allowed – if the node is given GCC conducted-mode permission, all restricted SI operations are allowed. If there is a peer SICE at the conducting node (i.e., one that has enrolled with the same session key), that SICE may also determine the ability to perform specific restricted operations by each other's peer SICE. If there is no peer SICE at that node, or if no peer SICE at that node chooses to act as an arbiter of SI privileges, only the GCC permission mechanism is used. If there is more than one SICE at the conducting node which is capable of acting as a conducting SICE, it is up to that node locally to determine which of these will take on the conducting function.

When a session switches to conducted mode, or if conductorship switches from one node to another, all SICEs are notified by receiving a GCC-Assign-Conductor indication from the GCC provider. This indication includes the GCC user ID of the node which has become the conductor. Each SICE may determine the user ID of the peer SICE at that node which has indicated itself as capable of arbitrating SI privileges. This is done by searching the current application roster for the entries corresponding to this GCC user ID and determining which, if any, of these has indicated itself as capable of arbitrating SI privileges. When a session switches to non-conducted mode, all SICEs are notified by receiving a GCC-Conductor-Release indication from the GCC provider.

When a session is first switched to conducted mode, or when a new conductor is assigned while already in conducted mode, no privileges are granted to any SICE to perform any restricted operation. While in conducted mode, if no GCC-Conductor-Permission-Grant indication has been received since the last change of conductorship or if the most recently received GCC-Conductor-Permission-Grant indication since the last change of conductorship had the permission flag set to FALSE, an SICE shall have no SI privileges unless specifically granted by the peer SICE at the conducting node. If the most recently received GCC-Conductor-Permission-Grant indication since the last change of conductorship had the permission flag set to TRUE, an SICE shall have all SI privileges granted.

If there is a peer SICE at the conducting node which has indicated itself as capable of arbitrating SI privileges, an SICE may request to be granted one or more privileges from the conductor. The following privileges may be requested:

- privilege to create, edit or delete a workspace;
- privilege to modify an annotation plane (a plane with the annotation usage designator set);
- privilege to modify an image plane (a plane with the image usage designator set);

- privilege to create, edit or delete a pointer;
- privilege to send remote keyboard or pointing device events;
- privilege to request remote printing;
- privilege to open an archive for writing or creation.

An SICE may request particular privileges or, by not including a privilege list in the request, may request all available privileges. Privileges are requested from the conductor by sending a ConductorPrivilegeRequestPDU to the conducting node. This is done in the manner described in Table 6-3. The content of the ConductorPrivilegeRequestPDU is shown in Table 8-70.

NOTE – The presence of a particular privilege allows only those actions which would otherwise have been available in non-conducted mode based on the negotiated capability set.

Table 8-70 – ConductorPrivilegeRequestPDU

Parameter	Description	
privilegeList	A set of values that indicate the privileges the requesting SICE wishes to be granted by the conductor.	
	workspacePrivilege:	This flag indicates that the requesting SICE wishes to be granted the privilege to create, edit or delete workspaces (although deletion is allowed for the reason of insufficientStorage).
	annotationPrivilege:	This flag indicates that the requesting SICE wishes to be granted the privilege to create, edit or delete annotation bitmaps or drawing elements on workspace planes designated for annotation usage.
	imagePrivilege:	This flag indicates that the requesting SICE wishes to be granted the privilege to create, edit or delete image bitmaps or video windows on workspace planes designated for image usage.
	pointingPrivilege:	This flag indicates that the requesting SICE wishes to be granted the privilege to create, edit or delete pointers.
	remoteKeyEventPrivilege:	This flag indicates that the requesting SICE wishes to be granted the privilege to send remote keyboard events. Note that permission from the workspace creator is still required before these events can be issued.
	remotePointingEventPrivilege:	This flag indicates that the requesting SICE wishes to be granted the privilege to send remote pointing device events. Note that permission from the workspace creator is still required before these events can be issued.
	remotePrintingPrivilege:	This flag indicates that the requesting SICE wishes to be granted the privilege to send remote printing requests.

Table 8-70 – ConductorPrivilegeRequestPDU

Parameter	Description	
	archiveCreateWritePrivilege:	This flag indicates that the requesting SICE wishes to be granted the privilege to open archives for creation or writing.
	nonStandardPrivilege:	This is a nonStandardIdentifier which represents a privilege understood by peer SICEs based on successful negotiation of the corresponding non-standard capability. The privilegeList may contain an arbitrary number of different nonStandardPrivileges.
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.	

On receipt of a ConductorPrivilegeRequestPDU, the SICE at the conducting node may grant some or all of the requested privileges by sending a ConductorPrivilegeGrantPDU to the requesting SICE. The ConductorPrivilegeGrantPDU includes the user ID of the SICE to which the privilege has been granted so that it can be broadcast to all other peer SICEs to inform them of the privilege granted to the requesting SICE. If no privileges are granted (beyond those that the requesting SICE may already have) no ConductorPrivilegeGrantPDU need be sent. If the SICE at the conducting node receives a GCC-Application-Roster-Report indication from the GCC provider and if the report indicates that new nodes have been added to the list of enrolled SICEs, the SICE at the conducting node shall re-broadcast the ConductorPrivilegeGrantPDU for each SICE which has some level of privilege. The SICE may at any time revoke some or all privileges or add privileges to any SICE by sending, unsolicited, a ConductorPrivilegeGrantPDU which indicates the new privilege list for the designated node in the same manner as used for granting privileges. The content of the ConductorPrivilegeGrantPDU is shown in Table 8-71.

Table 8-72 shows the effect of each of the privileges on the ability to transmit each SIPDU. The presence or absence of a privilege has no effect on the operation of receivers. If an SICE receives a PDU from another SICE that does not have the privilege to transmit that PDU, it shall process it as normal. A session refresher may initiate transactions needed to perform its function without any SI or GCC conductor privileges.

NOTE – The above restriction requiring receivers to process PDUs, even if they were in violation of their conducted mode privilege, is necessary to avoid race conditions during transitions when privileges are granted or removed.

Table 8-71 – ConductorPrivilegeGrantPDU

Parameter	Description	
destinationUserID	MCS user ID of the SICE to which the privileges are being granted or revoked.	
privilegeList	A set of values that indicate the privileges granted to an SICE by the conductor.	
	workspacePrivilege:	This flag indicates that the destination SICE has been granted the privilege to create, edit or delete workspaces (although deletion is allowed for the reason of insufficientStorage).
	annotationPrivilege:	This flag indicates that the destination SICE has been granted the privilege to create, edit or delete annotation bitmaps or drawing elements on workspace planes designated for annotation usage.
	imagePrivilege:	This flag indicates that the destination SICE has been granted the privilege to create, edit or delete image bitmaps and video windows on workspace planes designated for image usage.
	pointingPrivilege:	This flag indicates that the destination SICE has been granted the privilege to create, edit or delete pointers.
	remoteKeyEventPrivilege:	This flag indicates that the destination SICE has been granted the privilege to send remote keyboard events. Note that permission from the workspace creator is still required before these events can be issued.
	remotePointingEventPrivilege:	This flag indicates that the destination SICE has been granted the privilege to send remote pointing device events. Note that permission from the workspace creator is still required before these events can be issued.
	remotePrintingPrivilege:	This flag indicates that the destination SICE has been granted the privilege to send remote printing requests.
	archiveCreateWritePrivilege:	This flag indicates that the destination SICE has been granted the privilege to open archives for creation or writing.
	nonStandardPrivilege:	This is a nonStandardIdentifier which represents a privilege understood by peer SICEs based on successful negotiation of the corresponding non-standard capability. The privilegeList may contain an arbitrary number of different nonStandardPrivileges.

Table 8-71 – ConductorPrivilegeGrantPDU

Parameter	Description
nonStandardParameters (optional)	An optional list of non-standard parameters allowed only if the corresponding non-standard capabilities are present in the negotiated capability set.

Table 8-72 – Conducted mode operation summary

SIPDU	Privilege required
ArchiveAcknowledgePDU	None.
ArchiveClosePDU	None.
ArchiveErrorPDU	None.
ArchiveOpenPDU	None required to open for reading. Archive create or write privilege flag to open for creation or writing.
BitmapAbortPDU	Annotation privilege, image privilege or pointer privilege depending on the bitmap destination when sent by a bitmap transmitter. None required when sent by a bitmap receiver.
BitmapCheckpointPDU	Annotation privilege, image privilege or pointer privilege depending on the bitmap destination.
BitmapCreatePDU	Annotation privilege, image privilege or pointer privilege depending on the bitmap destination.
BitmapCreateContinuePDU	Annotation privilege, image privilege or pointer privilege depending on the bitmap destination.
BitmapDeletePDU	Annotation privilege, image privilege or pointer privilege depending on the bitmap destination.
BitmapEditPDU	Annotation privilege, Image Privilege, or Pointer Privilege depending on the bitmap destination.
ConductorPrivilegeGrantPDU	Only allowed by the conductor.
ConductorPrivilegeRequestPDU	None required.
DrawingCreatePDU	Annotation privilege.
DrawingDeletePDU	Annotation privilege.
DrawingEditPDU	Annotation privilege.
FontPDU	*FFS.*
RemoteEventPermissionGrantPDU	None required.
RemoteEventPermissionRequestPDU	Remote keyboard or pointing device privilege.
RemoteKeyboardEventPDU	Remote keyboard or pointing device privilege.
RemotePointingDeviceEventPDU	Remote keyboard or pointing device privilege.
RemotePrintPDU	Remote printing privilege.
SINonStandardPDU	Not defined by this Recommendation.
TextCreatePDU	*FFS.*
TextDeletePDU	*FFS.*
TextEditPDU	*FFS.*

Table 8-72 – Conducted mode operation summary

SIPDU	Privilege required
VideoWindowCreatePDU	Image privilege.
VideoWindowDeletePDU	Image privilege.
VideoWindowEditPDU	Image privilege.
WorkspaceCreatePDU	Workspace privilege.
WorkspaceCreateAcknowledgePDU	None required.
WorkspaceDeletePDU	Workspace privilege, although none required if reason is insufficientStorage.
WorkspaceEditPDU	Workspace privilege.
WorkspacePlaneCopyPDU	Annotation privilege, image privilege or both, depending on the usage designator of the destination plane.
WorkspaceReadyPDU	None required.
WorkspaceRefreshStatusPDU	None required.

9 SIPDU definitions

Each SIPDU is transported as one MCSSDU across an MCS connection. A standard ASN.1 data value encoding is used to transfer SIPDUs between peer SICEs. For all PDUs, the BASIC ALIGNED variant of the packed encoding rules of [ITU-T X.691] shall be used.

```
-- Begin SI Definitions
SI-PROTOCOL {itu-t(0) recommendation(0) t(20) t126(126) version(0) 2
asn1Modules(2) sI-PROTOCOL(1)} DEFINITIONS AUTOMATIC TAGS ::=
BEGIN
-- NOTE: All abstract types defined shall be exported.
-- ArchiveEntryName
   -- Name used to reference an archive entry.
ArchiveEntryName ::= BMPString (SIZE (1..256))

-- ArchiveError
   -- Specifies the cause of an error at a remote terminal during
   -- a workspace archive operation.
ArchiveError ::= CHOICE
{
    entryNotFound                NULL,
                                     -- The terminal does not have the entry that matches the
                                     -- archive name being accessed for reading, editing, or
                                     -- deletion.
    entryExists                  NULL,
                                     -- The terminal already has an archive entry that matches
                                     -- the name of the archive entry name being created.
    storageExceeded              NULL,
                                     -- The terminal does not have sufficient memory to store
                                     -- the requested information.
    archiveNoLongerAvailable     NULL,
                                     -- The archive indicated is no longer available.
    unspecifiedError             NULL,
                                     -- A general error that is not previously defined has occurred.
    nonStandardError             NonStandardIdentifier,
                                     -- Non-standard error code.
    ...
}
```

```
-- ArchiveHeader
   -- This type specifies the parameters used to address archives
   -- stored at remote terminals.
ArchiveHeader ::= SEQUENCE
{
    archiveName                 ArchiveName,
                                    -- Name of the archive.
    archiveCreationTime         GeneralizedTime,
                                    -- Time and date of the creation of the archive.
    archiveModificationTime     GeneralizedTime,
                                    -- Time and date of the most recent modification of the archive.
    ...
}

-- ArchiveMode
   -- One of the following sets of access modes must be indicated
   -- when an archive is opened.
ArchiveMode ::= SEQUENCE
{
    create      BOOLEAN,
                    -- TRUE indicates that the archive shall be created.  If an archive with
                    -- the same name exists, the operation should fail.
    read        BOOLEAN,
                    -- TRUE indicates that the archive shall be opened for reading only.
    write       BOOLEAN,
                    -- TRUE indicates that the archive shall be opened for writing.
    ...
}

-- ArchiveName
   -- Name used to reference an archive.
ArchiveName ::= BMPString (SIZE (1..256))
-- ArchiveOpenResult
   -- Specifies the result of an archive open request.
ArchiveOpenResult ::= CHOICE
{
    archiveOpenSuccessful       NULL,
                                    -- The requested archive was successfully opened.
    archiveNotFound             NULL,
                                    -- An archive to be opened for reading or writing was
                                    --  not found to exist.
    archiveTimeIncorrect        ArchiveHeader,
                                    -- An archive to be opened for reading or writing was found,
                                    --  but with incorrect creation or modification time. The
                                    -- actual header is included in the error response in this case.
    archiveExists               NULL,
                                    -- An archive to be opened for creation already exists and will
                                    -- not be overwritten.
    archiveOpenForWriting       NULL,
                                    -- An archive to be opened for writing is already open for writing.
    storageExceeded             NULL,
                                    -- The terminal does not have sufficient memory to store the
                                    -- requested archive.
    unspecifiedError            NULL,
                                    -- An unspecified error has occurred preventing the archive
                                    -- from being opened.
    nonStandardResult           NonStandardIdentifier,
                                    -- Non-standard result code.
    ...
}

-- BitmapAbortReason
   -- These values represent the possible reason codes
   -- for the BitmapAbortPDU.
BitmapAbortReason ::= CHOICE
{
```

```
    unspecified                     NULL,
                                        -- Bitmap aborted for an unspecified reason.
    noResources                     NULL,
                                        -- Bitmap creation failed due to local resource management
                                        -- problems.
    outOfPaper                      NULL,
                                        -- Bitmap creation failed because the receiving terminal is
                                        -- out of paper.
    nonStandardReason               NonStandardParameter,
    ...
}

-- BitmapAttribute
   -- This CHOICE represents the list of possible bitmap attributes.
BitmapAttribute ::= CHOICE
{
    viewState                       ViewState,
                                        -- Indicates the state.
    zOrder                          ZOrder,
                                        -- Used to set the bitmap to the front or back of the display
                                        --  list within an addressable plane.
    nonStandardAttribute            NonStandardParameter,
    ...,

-- Parameters added during 1st revision
transparencyMask                    TransparencyMask
}

-- BitmapData
   -- All or part of a bitmap bitstream.
BitmapData ::= SEQUENCE
{
    dataCheckpoint                  SEQUENCE (SIZE (1..100)) OF TokenID OPTIONAL,
                                        -- Tokens to uninhibit when the corresponding data is ready
                                        -- for display if checkpointing is enabled for the exchange
    padBits                         INTEGER (1..256) OPTIONAL,
                                        -- Count of bits at the end of the data octets that are not part
                                        -- of the image bitstream and are to be ignored
    data                            OCTET STRING (SIZE (1..8192)),
                                        -- The compression-format-specific bitmap data.
    ...
}

-- BitmapDestinationAddress
   -- Destination address for bitmap exchanges.
BitmapDestinationAddress ::= CHOICE
{
    hardCopyDevice                  NULL,
    softCopyImagePlane              SoftCopyDataPlaneAddress,
    softCopyAnnotationPlane         SoftCopyDataPlaneAddress,
    softCopyPointerPlane            SoftCopyPointerPlaneAddress,
    ...,

-- Parameters added during 1st revision
nonStandardDestination              NonStandardParameter
}

-- BitmapHeaderUncompressed
   -- This type specifies the parameters of uncompressed bitmap
   -- bitstreams.
BitmapHeaderUncompressed ::= SEQUENCE
{
    colorMappingMode                CHOICE
    {
        directMap                   SEQUENCE
        {
            colorSpace                  ColorSpaceSpecifier,
```

```
            resolutionMode              ColorResolutionModeSpecifier
        },
        paletteMap                  SEQUENCE
        {
            colorPalette                ColorPalette,
            bitsPerPixel                INTEGER (1 | 4 | 8)
         },
         ...
    },
    ...
}

-- BitmapHeaderT4
   -- Bitmap header for T.4 (G3) encoding
BitmapHeaderT4 ::= SEQUENCE
{
    twoDimensionalEncoding          BOOLEAN,
                                        -- 2-D encoding if TRUE,
                                        -- 1-D encoding if FALSE
    ...
}

-- BitmapHeaderT6
   -- Bitmap header for T.6 (G4) encoding
BitmapHeaderT6 ::= SEQUENCE
{
    ...
}

-- BitmapHeaderT81
   -- This type is used to specify the parameters necessary to
   -- decode and display a T.81 (JPEG) image that are not specified
   -- within the T.81 bitstream.
BitmapHeaderT81 ::=  SEQUENCE
{
    colorSpace                      ColorSpaceSpecifier,
    resolutionMode                  ColorResolutionModeSpecifier,
    ...,

-- Parameters added during 1st revision
colorPalette                        ColorPalette OPTIONAL
                                        -- Color palette to be optionally used by the receiver to render
                                        -- the associated bitmap if the local display device is
                                        -- palette-mapped. This parameter is provided as a
                                        -- convenience for receiver rendering.
}

-- BitmapHeaderT82
   -- This type is used to specify the parameters necessary to
   -- decode and display a T.82 (JBIG) image that are not specified
   -- within the T.82 bitstream.
BitmapHeaderT82 ::=  SEQUENCE
{
    colorMappingMode                CHOICE
    {
        directMap                   ColorSpaceSpecifier,
                                        -- Only greyscale and RGB colorspaces are allowed.
        paletteMap                  SEQUENCE
        {
            bitmapPalette               ColorPalette,
            progressiveMode             CHOICE
            {
                progressivePalettes         SEQUENCE (SIZE (1..8)) OF ColorIndexTable,
                selfProgressive             NULL,
                ...
            } OPTIONAL
        }
    },
```

```
    ...
}

-- BitmapRegion
   -- This type specifies a rectangular subregion within a bitmap.
BitmapRegion ::= SEQUENCE
{
    upperLeft                   SEQUENCE
    {
        xCoordinate                 INTEGER (0..65535),
                                        -- X component of a Cartesian address
        yCoordinate                 INTEGER (0..65535)
                                        -- Y component of a Cartesian address
    },
    lowerRight                  SEQUENCE
    {
        xCoordinate                 INTEGER (0..65535),
                                        -- X component of a Cartesian address
        yCoordinate                 INTEGER (0..65535)
                                        -- Y component of a Cartesian address
    }
}

-- BitmapSize
   -- The size of a bitmap in pixels.
BitmapSize ::= SEQUENCE
{
    width                       INTEGER (1..65536),
                                    -- The number of pixels horizontally
    height                      INTEGER (1..65536)
                                    -- The number of pixels vertically
}

-- ButtonEvent
   -- Describes pointing device button events.
ButtonEvent ::= CHOICE
{
    buttonUp                    NULL,
                                    -- The button is up.
    buttonDown                  NULL,
                                    -- The button is down.
    buttonDoubleClick           NULL,
                                    -- A button down event occurred within the double-click
                                    -- time window.
    buttonTripleClick           NULL,
                                    -- A button down event occurred within the triple-click
                                    -- time window.
    buttonQuadClick             NULL,
                                    -- A button down event occurred within the quad-click
                                    -- time window.
    nonStandardButtonEvent      NonStandardIdentifier,
    ...
}

-- ColorAccuracyEnhancementCIELab
ColorAccuracyEnhancementCIELab ::= CHOICE
{
    predefinedCIELabSpace           CHOICE
    {
        nonStandardCIELabSpace      NonStandardParameter,
        ...
    },
    generalCIELabParameters         SEQUENCE
    {
        colorTemperature            INTEGER (0..MAX) OPTIONAL,
                                        -- Color temperature of the white point assumed by the color
                                        -- space (in degrees Kelvin)
```

```
        gamut                   SEQUENCE
        {
            lSpan               INTEGER (-32768..32767),
                                    -- max L* – min L*
            lOffset             INTEGER (-32768..32767),
                                    -- offset of the zero point for L
            aSpan               INTEGER (-32768..32767),
                                    -- max a* – min a*
            aOffset             INTEGER (-32768..32767),
                                    -- offset of the zero point for a
            bSpan               INTEGER (-32768..32767),
                                    -- max b* – min b*
            bOffset             INTEGER (-32768..32767)
                                    -- offset of the zero point for b
        } OPTIONAL,
        ...
    },
    ...
}

-- ColorAccuracyEnhancementGreyscale
ColorAccuracyEnhancementGreyscale ::= CHOICE
{
    predefinedGreyscaleSpace    CHOICE
    {
        nonStandardGreyscaleSpace   NonStandardParameter,
        ...
    },
    generalGreyscaleParameters  SEQUENCE
    {
        gamma                   REAL (0..MAX) OPTIONAL,
                                    -- Gamma value of the color space
        ...
    },
    ...
}

-- ColorAccuracyEnhancementRGB
ColorAccuracyEnhancementRGB ::= CHOICE
{
    predefinedRGBSpace          CHOICE
    {
        nonStandardRGBSpace     NonStandardParameter,
        ...
    },
    generalRGBParameters        SEQUENCE
    {
        gamma                   REAL (0..MAX) OPTIONAL,
                                    -- Gamma value of the color space
        colorTemperature        INTEGER (0..MAX) OPTIONAL,
                                    -- Color temperature of the white point assumed by the color
                                    -- space (in degrees Kelvin)
        primaries               SEQUENCE
        {
            red             ColorCIExyChromaticity,
                                -- CIE xy chromaticity coordinate of the red primary
            green           ColorCIExyChromaticity,
                                -- CIE xy chromaticity coordinate of the green primary
            blue            ColorCIExyChromaticity
                                -- CIE xy chromaticity coordinate of the blue primary
        } OPTIONAL,
        ...
    },
    ...
}

-- ColorAccuracyEnhancementYCbCr
ColorAccuracyEnhancementYCbCr ::= CHOICE
```

```
{
    predefinedYCbCrSpace            CHOICE
    {
        cCIR709                     NULL,
        nonStandardRGBSpace         NonStandardParameter,
        ...
    },
    generalYCbCrParameters          SEQUENCE
    {
        gamma                       REAL (0..MAX) OPTIONAL,
                                        -- Gamma value of the color space
        colorTemperature            INTEGER (0..MAX) OPTIONAL,
                                        -- Color temperature of the white point  assumed by the color
                                        --  space (in degrees Kelvin)
        primaries                   SEQUENCE
        {
            red                     ColorCIExyChromaticity,
                                        -- CIE xy chromaticity coordinate of the red primary
            green                   ColorCIExyChromaticity,
                                        -- CIE xy chromaticity coordinate of the green primary
            blue                    ColorCIExyChromaticity
                                        -- CIE xy chromaticity coordinate of the blue primary
        } OPTIONAL,
        ...
    },
    ...
}

-- ColorCIELab
   -- Definition of a CIELab color.
ColorCIELab ::= SEQUENCE
{
    l       INTEGER (0..255),
                -- Perceptually normalized luminance component
    a       INTEGER (0..255),
                -- One of two perceptually normalized chroma components
    b       INTEGER (0..255)
                -- One of two perceptually normalized chroma components
}

-- ColorCIExyChromaticity
   -- Definition of a CIE normalized chromaticity value.
ColorCIExyChromaticity ::= SEQUENCE
{
    x       REAL (0..one),
                -- CIE normalized x component
    y       REAL (0..one)
                -- CIE normalized y component
}

-- ColorIndexTable
   -- This type is used to specify collections of color
   -- values.  All entries are references to absolute
   -- color palette data.
ColorIndexTable ::= SEQUENCE (SIZE (1..256)) OF INTEGER (0..255)

-- ColorPalette
ColorPalette    ::= SEQUENCE
{
    colorLookUpTable            CHOICE
    {
        paletteRGB              SEQUENCE
        {
            palette             SEQUENCE (SIZE (2..256)) OF ColorRGB,
            enhancement         ColorAccuracyEnhancementRGB OPTIONAL,
            ...
        },
        paletteCIELab           SEQUENCE
```

```
            {
                palette                 SEQUENCE (SIZE (2..256)) OF ColorCIELab,
                enhancement             ColorAccuracyEnhancementCIELab OPTIONAL,
                ...
            },
            paletteYCbCr                SEQUENCE
            {
                palette                 SEQUENCE (SIZE (2..256)) OF ColorYCbCr,
                enhancement             ColorAccuracyEnhancementYCbCr OPTIONAL,
                ...
            },
            nonStandardPalette          NonStandardParameter,
            ...
        },
        transparentEntry                INTEGER (0..255)            OPTIONAL,
                                            -- Index value of transparent color
        ...
}

-- ColorResolutionModeSpecifier
ColorResolutionModeSpecifier ::= CHOICE
{
        resolution4-4-4                 NULL,
                                            -- Indicates single component
                                            -- 4:4:4
        resolution-4-2-2                NULL,
                                            -- 4:2:2 chrominance sub-sampling
        resolution-4-2-0                NULL,
                                            -- 4:2:0 chrominance sub-sampling
        nonStandardResolutionMode       NonStandardIdentifier,
        ...
}

-- ColorRGB
    -- Definition of an RGB color.
ColorRGB ::= SEQUENCE
{
        r       INTEGER (0..255),
                    -- Red color component
        g       INTEGER (0..255),
                    -- Green color component
        b       INTEGER (0..255)
                    -- Blue color component
}

-- ColorSpaceSpecifier
ColorSpaceSpecifier ::=   CHOICE
{
        greyscale               SEQUENCE
        {
            accuracyEnhancement         ColorAccuracyEnhancementGreyscale OPTIONAL
        },
        yCbCr                   SEQUENCE
        {
            accuracyEnhancement         ColorAccuracyEnhancementYCbCr OPTIONAL
        },
        rgb                     SEQUENCE
        {
            accuracyEnhancement         ColorAccuracyEnhancementRGB OPTIONAL
        },
        cieLab                  SEQUENCE
        {
             accuracyEnhancement        ColorAccuracyEnhancementCIELab OPTIONAL
        },
        nonStandardColorSpace           NonStandardIdentifier,
        ...
}
```

```
-- ColorYCbCr
   -- Definition of a YCbCr color.
ColorYCbCr ::= SEQUENCE
{
    y       INTEGER (0..255),
                -- Luminance component
    cb      INTEGER (0..255),
                -- Normalized blue minus luminance component
    cr      INTEGER (0..255)
                -- Normalized red minus luminance component
}

-- ConductorPrivilege
   -- List of privileges that are awarded by the SICE at the conducting
   -- node to other SICEs in the session.
ConductorPrivilege ::= CHOICE
{
    workspacePrivilege              NULL,
                                        -- Privilege to create, edit, or delete workspaces
    annotationPrivilege             NULL,
                                        -- Privilege to create, edit, or delete annotation bitmaps
                                        -- or drawing elements
    imagePrivilege                  NULL,
                                        -- Privilege to create, edit, or delete image bitmaps
    pointingPrivilege               NULL,
                                        -- Privilege to create, edit, or delete pointers
    remoteKeyEventPrivilege         NULL,
                                        -- Privilege to send remote keyboard events
    remotePointingEventPrivilege    NULL,
                                        -- Privilege to send pointing device events
    remotePrintingPrivilege         NULL,
                                        -- Privilege to request remote printing
    archiveCreateWritePrivilege     NULL,
                                        -- Privilege to create or append an archive
    nonStandardPrivilege            NonStandardIdentifier,
                                        -- Non-standard privilege that was successfully negotiated.
    ...
}

-- DataPlaneID
   -- This is the identifier of a data plane within a workspace.
DataPlaneID ::= INTEGER (0..255)

-- DrawingAttribute
   -- The following drawingAttributes are used to specify visual and
   -- behavioral properties of a drawing.
DrawingAttribute ::= CHOICE
{
    penColor                        WorkspaceColor,
                                        -- Color of drawing pen
    fillColor                       WorkspaceColor,
                                        -- Color used to fill a closed region
    penThickness                    PenThickness,
                                        -- Width of pen
    penNib                          PenNib,
                                        -- Shape of pen nib
    lineStyle                       LineStyle,
                                        -- Style of line
    highlight                       BOOLEAN,
                                        -- Flag indicating whether the drawing element should be
                                        -- of a solid color or a highlight (semi-transparent)
    viewState                       ViewState,
                                        -- Indicates the visibility state
    zOrder                          ZOrder,
                                        -- Used to set the graphical element to the front or back of the
                                        -- display list within an addressable plane
```

```
    nonStandardAttribute            NonStandardParameter,
    ...
}

-- DrawingDestinationAddress
   -- A DrawingDestinationAddress specifies the destination of drawing
   -- elements.
DrawingDestinationAddress ::= CHOICE
{
    softCopyAnnotationPlane         SoftCopyDataPlaneAddress,
    ...,
-- Parameters added during 1st revision
nonStandardDestination              NonStandardParameter
}

-- DrawingType
   -- A DrawingType specifies the shape of a drawn element.
DrawingType ::= CHOICE
{
    point                           NULL,
                                        -- Unconnected points
    openPolyLine                    NULL,
                                        -- Points connected with straight lines. The last point is not
                                        -- connected to the first.
    closedPolyLine                  NULL,
                                        -- Points connected with straight lines.
                                        -- The last point is connected to the first.
    rectangle                       NULL,
                                        -- A rectangle defined by two corners
    ellipse                         NULL,
                                        -- An ellipse
    nonStandardDrawingType          NonStandardIdentifier,
                                        -- Negotiated non-standard type
    ...
}
DSMCCTap ::= SEQUENCE
{
     use                            INTEGER (0..65535),
                                        -- the use for the Tap
     id                             INTEGER (0.. 65535),
                                        -- identifier for the Tap
     associationTag                 INTEGER (0..65535),
                                        -- group identifier for Tap resource descriptors
     selector                       OCTET STRING (SIZE (1..256)) OPTIONAL,
                                        --  upper protocol selector info
     ...
}

-- EditablePlaneCopyDescriptor
   -- Paired list of handles for source objects and their copies.
EditablePlaneCopyDescriptor ::= SEQUENCE
{
     objectList                     SEQUENCE (SIZE (1..65536)) OF SEQUENCE
     {
          sourceObjectHandle        Handle,
          destinationObjectHandle   Handle
                                        -- This handle is used to reference the new copy of the
                                        -- source object in the future.
     },
     destinationOffset              WorkspacePoint OPTIONAL,
                                        -- This parameter defines an offset to be added to the
                                        -- coordinates of all of the copied objects. If not present,
                                        -- zero offset is assumed.
     planeClearFlag                 BOOLEAN,
                                        -- When FALSE, the destination objects are appended to
                                        -- the existing set of objects in the destination plane. When
                                        -- TRUE, all existing objects in the destination plane are
```

```
                                        -- deleted prior to the copy operation.
    ...
}

-- Handle
   -- Unique identifier that is used to address objects to allow edit
   -- and/or delete operations. These are obtained from GCC via the
   -- GCC-Registry-Allocate-Handle request/confirm primitives.
Handle ::= INTEGER (0..4294967295)

-- KeyCode
   -- Character code that is contained in a RemoteKeyboardEventPDU.
   -- This is either a two-octet value that uses the UNICODE character
   -- representation or special key specifier.
KeyCode ::= CHOICE
{
    character                   BMPString (SIZE (1)),
                                    -- UNICODE character
    fkey                        INTEGER (1..32),
                                    -- Function key

                                    -- Edit and navigation keys
    upArrow                     NULL,
    downArrow                   NULL,
    leftArrow                   NULL,
    rightArrow                  NULL,
    pageUp                      NULL,
    pageDown                    NULL,
    home                        NULL,
    end                         NULL,
    insert                      NULL,
    delete                      NULL,

    nonStandardKey              NonStandardIdentifier,
                                    -- Non-standard key code
    ...
}

-- KeyModifier
   -- Collection of keyboard modifiers.
KeyModifier ::= CHOICE
{
    leftAlt                     NULL,
                                    -- Indicates the left ALT modifier key is pressed
    rightAlt                    NULL,
                                    -- Indicates the right ALT modifier key is pressed
    leftShift                   NULL,
                                    -- Indicates the left SHIFT modifier key is pressed
    rightShift                  NULL,
                                    -- Indicates the right SHIFT modifier key is pressed
    leftControl                 NULL,
                                    -- Indicates the left CONTROL modifier key is pressed
    rightControl                NULL,
                                    -- Indicates the right CONTROL modifier key is pressed
    leftSpecial                 NULL,
                                    -- Indicates the left SPECIAL modifier key is pressed
    rightSpecial                NULL,
                                    -- Indicates the right SPECIAL modifier key is pressed
    numberPad                   NULL,
                                    -- Indicates the associated keystroke is actuated by the
                                    -- numeric keypad
    scrollLock                  NULL,
                                    -- Indicates that the scroll lock is active
    nonStandardModifier         NonStandardIdentifier,
                                    -- Non-standard key modifier.
    ...
}
```

```
-- KeyPressState
   -- Set of events for a key, used as part of
   -- a RemoteKeyboardEventPDU
KeyPressState ::= CHOICE
{
    none                        NULL,
                                    -- No key event is signaled. This is used when only keyboard
                                    -- modifier keys are changing state.
    keyPress                    NULL,
                                    -- A key press event has occurred. Note that multiple keyPress
                                    -- events may occur as a result of a keyDown event and keyboard
                                    -- auto-repeat.
    keyDown                     NULL,
                                    -- A key down transition has occurred. Note that this implies a
                                    -- keyUp for an unmatched previously received keyDown.
    keyUp                       NULL,
                                    -- A key up transition has occurred.
    nonStandardKeyPressState    NonStandardIdentifier,
                                    -- Non-standard key press state.
    ...
}

-- LineStyle
   -- The LineStyle attribute is used during a line draw procedure. It
   -- specifies the type of line drawn.
LineStyle ::= CHOICE
{
    solid                       NULL,
                                    -- All pixels between endpoints are to be drawn.
    dashed                      NULL,
                                    -- A dashed pattern is to be applied.
    dotted                      NULL,
                                    -- A dotted pattern is to be applied.
    dash-dot                    NULL,
                                    -- A dash-dot pattern is to be applied.
    dash-dot-dot                NULL,
                                    -- A dash-dot-dot pattern is to be applied.
    two-tone                    NULL,
                                    -- Line color is to be applied to 50% of the line width with a
                                    -- complimentary color applied to either side. The width of
                                    -- either side region is to be 25% of the line width.
    nonStandardStyle            NonStandardIdentifier,
    ...
}

-- MCSUserID
   -- This type is used to specify MCS User IDs.
MCSUserID ::= INTEGER (1001..65535)

-- H221NonStandardIdentifier
   -- Used to specify non-standard objects using H.221 numbering.
   -- The first four octets shall designate country code and
   -- manufacturer code, assigned as specified in
   -- Annex A/H.221, for NS-cap and NS-comm.
H221NonStandardIdentifier ::= OCTET STRING (SIZE (4..255))

-- NonStandardIdentifier
   -- Unique identifier used to specify non-standard capabilities and
   -- parameters either as an ASN.1 OBJECT IDENTIFIER or as an H.221
   -- non-standard object.
NonStandardIdentifier ::= CHOICE
{
    object                      OBJECT IDENTIFIER,
    h221nonStandard             H221NonStandardIdentifier
}
```

```
-- NonStandardParameter
   -- Used to specify non-standard parameters.  This includes a
   -- data field which may be used to fill in parameter values
   -- of the type indicated by the NonStandardIdentifier.
NonStandardParameter ::= SEQUENCE
{
    nonStandardIdentifier               NonStandardIdentifier,
    data                                OCTET STRING
}

-- one
   -- This type provides a real value = 1 for use in this Recommendation.
one REAL ::= {mantissa 1,  base 2,  exponent 0}

-- PenNib
   -- This type specifies the shape of the nib of the pen that is
   -- used to draw graphical elements.
PenNib ::= CHOICE
{
    circular                            NULL,
                                            -- A circle is used for the nib shape.
    square                              NULL,
                                            -- A square is used for the nib shape.
    nonStandardNib                      NonStandardIdentifier,
                                            -- A non-standard pen nib
    ...
}

-- PenThickness
   -- This type specifies the thickness of the pen that is used to
   -- draw graphical elements.
PenThickness ::= INTEGER (1..255)

-- PermanentPlaneCopyDescriptor
   -- Describes source and destination regions within the corresponding
   -- planes to be copied from and to. This is only to be used when the
   -- source and destination planes are permanent.
PermanentPlaneCopyDescriptor ::= SEQUENCE
{

     sourceRegion                       WorkspaceRegion,
                                            -- Source rectangle to be copied.
     destinationRegion                  WorkspaceRegion,
                                            -- Destination rectangle to be copied.
                                            -- May be restricted by caps to be the same size as
                                            -- the source region.
     ...
}

-- PixelAspectRatio
   -- This type specifies that horizontal to vertical ratio of
   -- the size of a pixel.
PixelAspectRatio ::= CHOICE
{
    square                              NULL,
                                            -- pixel aspect ratio is 1:1
    cif                                 NULL,
                                            -- pixel aspect ratio is 12:11 (hor:ver)
    fax1                                NULL,
                                            -- 385:800 (hor:ver)
                                            -- 8 lines/mm horizontally,
                                            -- 3.85 lines/mm vertically
    fax2                                NULL,
                                            -- 770:800 (hor:ver)
                                            -- 8 lines/mm horizontally,
```

```
                                        -- 7.7 lines/mm vertically
    general                             SEQUENCE
                                        -- The following two integers specify a rational number that
                                        -- is equivalent to a pixel's width divided by a pixel's height.
    {
        numerator                   INTEGER (1..65535),
        denominator                 INTEGER (1..65535)
    },
    nonStandardAspectRatio          NonStandardIdentifier,
    ...
}

-- PlaneAttribute
   -- Plane attributes are editable characteristics of workspace planes.
PlaneAttribute ::= CHOICE
{
    protection                      PlaneProtection,
                                        -- Access restrictions for a plane
    nonStandardAttribute            NonStandardParameter,
                                        -- Non-standard attribute
    ...
}

-- PlaneProtection
   -- This enumeration identifies the possible access restrictions
   -- that can be imposed on a workspace plane.
PlaneProtection ::= SEQUENCE
{
    protected                       BOOLEAN,
                                        -- Only the SICEs granted access via the
                                        -- protectedPlaneAccessList can submit data to this plane.
    ...
}

-- PlaneUsage
   -- This type specifies the usage of a single plane in a workspace.
PlaneUsage ::= CHOICE
{
    annotation                      NULL,
                                        -- The plane is designated to contain annotation data.
    image                           NULL,
                                        -- The plane is designated to contain image data.
    nonStandardPlaneUsage           NonStandardIdentifier,
                                        -- The plane is designated to contain non-standard plane data.
    ...
}

-- PointList
   -- A list of points to define a drawing object using one of
   -- three possible encodings depending on how far any point
   -- in the list strays from the anchor point.
PointList ::= CHOICE
{
    pointsDiff4     SEQUENCE (SIZE (0..255)) OF PointDiff4,
    pointsDiff8     SEQUENCE (SIZE (0..255)) OF PointDiff8,
    pointsDiff16    SEQUENCE (SIZE (0..255)) OF PointDiff16
}

-- PointListEdits
   -- A list of points to edit a drawing object using one of
   -- three possible encodings depending on how far any point
   -- in the list strays from the anchor point.
PointListEdits ::= SEQUENCE SIZE (1..255) OF SEQUENCE
{
    initialIndex                    INTEGER (0..65534),
                                        -- Index of the first (or only) point to edit
    initialPointEdit                PointDiff16,
```

```
                                        -- Position of the point specified relative to the anchor point
    subsequentPointEdits                PointList OPTIONAL,
                                        -- Points specified relative to the previous point in this list
                                        -- (the first one specified relative to the initialPointEdit).
                                        -- When this list is used, successive points are assumed to
                                        -- have indices sequentially following the initialPointIndex.
    ...
}

-- PointDiff4
   -- A point specified differentially relative to an anchor point
   -- with a range from –8 to +7.
PointDiff4 ::= SEQUENCE
{
    xCoordinate                         INTEGER (-8..7),
                                        -- X component of a Cartesian address
    yCoordinate                         INTEGER (-8..7)
                                        -- Y component of a Cartesian address
}

-- PointDiff8
   -- A point specified differentially relative to an anchor point
   -- with a range from –128 to +127.
PointDiff8 ::= SEQUENCE
{
    xCoordinate                         INTEGER (-128..127),
                                        -- X component of a Cartesian address
    yCoordinate                         INTEGER (-128..127)
                                        -- Y component of a Cartesian address
}

-- PointDiff16
   -- A point specified differentially relative to an anchor point
   -- with a range from –32768 to +32767.
PointDiff16 ::= SEQUENCE
{
    xCoordinate                         INTEGER (-32768..32767),
                                        -- X component of a Cartesian address
    yCoordinate                         INTEGER (-32768..32767)
                                        -- Y component of a Cartesian address
}

-- RemoteEventDestinationAddress
   -- A RemoteEventDestinationAddress specifies the destination of
   -- a remote event.
RemoteEventDestinationAddress ::= CHOICE
{
    softCopyWorkspace                   Handle,
    ...,

-- Parameters added during 1st revision
nonStandardDestination                  NonStandardParameter
}

-- RemoteEventPermission
   -- Choice of remote events that can be issued to a workspace.
RemoteEventPermission ::= CHOICE
{
    keyboardEvent                       NULL,
    pointingDeviceEvent                 NULL,
    nonStandardEvent                    NonStandardIdentifier,
    ...
}

-- RotationSpecifier
   -- Specifies a rotation angle and an axis of revolution
RotationSpecifier ::= SEQUENCE
```

```
{
    rotationAngle                   INTEGER (0..21599),
                                        -- 0 degrees to 359 degrees 59 minutes in units of minutes
                                        -- of arc.
    rotationAxis                    PointDiff16
                                        -- Workspace location relative to an object's anchor point.
}

-- SoftCopyDataPlaneAddress
   -- Address of a workspace data plane.
SoftCopyDataPlaneAddress ::= SEQUENCE
{
    workspaceHandle                 Handle,
    plane                           DataPlaneID
}

-- SoftCopyPointerPlaneAddress
   -- Address of a workspace pointer plane.
SoftCopyPointerPlaneAddress ::= SEQUENCE
{
    workspaceHandle                 Handle
}

-- SourceDisplayIndicator
   -- Indicator of the size and location of a workspace view within
   -- the display device of the sourcing terminal.
SourceDisplayIndicator ::= SEQUENCE
{
    displayAspectRatio              REAL (0..MAX),
                                        -- Aspect ratio of the display; horizontal over vertical size.
                                        -- Positive real values.
    horizontalSizeRatio             REAL (0..MAX),
                                        -- Ratio of workspace view horizontal dimension to display
                                        -- horizontal dimension.
                                        -- Positive real values.
    horizontalPosition              REAL,
                                        -- Horizontal offset of upper left corner of the workspace view
                                        -- from the upper left corner of the display normalized to the
                                        -- display width (where the display spans the horizontal
                                        -- range 0.0 to 1.0).
    verticalPosition                REAL,
                                        -- Vertical offset of upper left corner of the workspace view
                                        -- from the upper left corner of the display normalized to
                                        -- the display height (where the display spans the vertical
                                        -- range 0.0 to 1.0).
    ...
}

-- TokenID
   -- MCS Token ID.
TokenID ::= INTEGER (1..65535)

-- TransparencyMask
   -- A binary bitmap that indicates which pixels in a bitmap shall be
   -- treated as transparent.
TransparencyMask ::= SEQUENCE
{
     bitMask                        CHOICE
     {
          uncompressed              OCTET STRING,
                                        -- Binary bitmap where a value of 1 indicates that the
                                        -- corresponding pixel in the reference bitmap shall be
                                        -- displayed. A value of 0 indicates that that pixel shall be
                                        -- treated as transparent.
          jbigCompressed            OCTET STRING,
                                        -- Same as above but additionally compressed using JBIG.
```

```
            nonStandardFormat               NonStandardParameter,
            ...
        },
        nonStandardParameters               SET OF NonStandardParameter OPTIONAL,
        ...
}

-- VideoWindowDestinationAddress
    -- A VideoWindowDestinationAddress specifies the destination of video windows.
VideoWindowDestinationAddress ::= CHOICE
{
        softCopyImagePlane                  SoftCopyDataPlaneAddress,
        nonStandardDestination              NonStandardParameter,
        ...
}

-- VideoSourceIdentifier
    -- Used to reference an out-of-band video source.
VideoSourceIdentifier ::= CHOICE
{
        default                             NULL,
        h243SourceIdentifier                OCTET STRING (SIZE (2)),
                                                -- A two-octet field. The first octet should contain
                                                -- the H.243 MCU ID (M), and the second octet should
                                                -- contain the H.243 Terminal ID (T).
        h245SourceIdentifier                INTEGER (0..65535),
        dSMCCConnBinder                     SEQUENCE OF DSMCCTap,
        videoIdentifier                     OCTET STRING (SIZE (1..256)),
        nonStandardSourceIdentifier         NonStandardParameter,
        ...
}

-- VideoWindowAttribute
    -- Attributes of video windows.
VideoWindowAttribute ::= CHOICE
{
        transparencyMask                    TransparencyMask,
                                                -- Bit mask specifying which pixels should be treated
                                                -- as transparent within the video window.
        nonStandardAttribute                NonStandardParameter,
        ...
}

-- VideoWindowCreatePDU
    -- This PDU allows video windows encapsulating out of band video
    -- streams to be created.
VideoWindowCreatePDU ::= SEQUENCE
{
        videoWindowHandle                   Handle,
                                                -- Handle to be used to reference this object in the future
        destinationAddress                  VideoWindowDestinationAddress,
                                                -- Destination address of the video window
        videoSourceIdentifier               VideoSourceIdentifier,
                                                -- Identifies the video source to be placed in the window
        attributes                          SET OF VideoWindowAttribute OPTIONAL,
                                                -- List of editable attributes of the video window
        anchorPoint                         WorkspacePoint OPTIONAL,
                                                -- Point of origin of the video window with respect to the
                                                -- destination workspace. Only needed for softcopy
                                                -- bitmaps. Default is (0,0).
        videoWindowSize                     BitmapSize,
                                                -- Width and height of the total video window represented
                                                -- in the bitstream.
        videoWindowRegionOfInterest         BitmapRegion OPTIONAL,
                                                -- Region of interest within the video stream to be applied
                                                -- to the workspace Default is full video area.
        pixelAspectRatio                    PixelAspectRatio,
```

```
                                        -- Pixel aspect ratio of the video stream
    scaling                             PointDiff16 OPTIONAL,
                                        -- Offset in workspace coordinates of the lower right hand
                                        -- corner of the video window relative to the anchor point
    nonStandardParameters               SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard
                                        -- capabilities are present in the negotiated capability set.
    ...
}

-- VideoWindowEditPDU
   -- A VideoWindowEditPDU is used to alter one or more of
   -- a video window element's attributes or parameters.
VideoWindowEditPDU ::= SEQUENCE
{
    videoWindowHandle                   Handle,
                                        -- Identifier of item to be edited
    videoSourceIdentifierEdit           VideoSourceIdentifier OPTIONAL,
                                        -- Identifies the video source to be placed in the window
    attributeEdits                      SET OF VideoWindowAttribute OPTIONAL,
                                        -- List of attribute changes
    anchorPointEdit                     WorkspacePoint OPTIONAL,
                                        -- Point of origin of the drawing element
    videoWindowSize                     BitmapSize,
                                        -- Change to the width and height of the total video window
                                        -- represented in the bitstream
    videoWindowRegionOfInterestEdit     BitmapRegion OPTIONAL,
                                        -- Change to the region of interest within the video stream
                                        -- to be applied to the workspace
    pixelAspectRatioEdit                PixelAspectRatio OPTIONAL,
                                        -- Change to the pixel aspect ratio of the video stream
    scalingEdit                         PointDiff16 OPTIONAL,
                                        -- Change to the offset in workspace coordinates of the
                                        -- lower right hand corner of the video window relative to
                                        -- the anchor point
    nonStandardParameters               SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- ViewState
   -- Controls the visibility state of an object.
ViewState ::= CHOICE
{
    unselected                          NULL,
    selected                            NULL,
    hidden                              NULL,
    nonStandardViewState                NonStandardIdentifier,
    ...
}

-- WorkspaceAttribute
   -- Workspace attributes are editable characteristics of workspace.
WorkspaceAttribute ::= CHOICE
{
    backgroundColor                     WorkspaceColor,
                                        -- This specifies the background color of the workspace.
    preserve                            BOOLEAN,
                                        -- If TRUE, the associated workspace resource should not
                                        -- be placed on the viewed workspace queue once it has been
                                        -- automatically removed from the Focus state.
    nonStandardAttribute                NonStandardParameter,
    ...
}
```

```
-- WorkspaceColor
   -- The following defines a generic type for a color, used where a
   -- color is required for drawing or workspace backgrounds.
WorkspaceColor ::= CHOICE
{
    workspacePaletteIndex           INTEGER (0..255),
    rgbTrueColor                    ColorRGB,
    transparent                     NULL,
    ...
}

-- WorkspaceCoordinate
   -- A WorkspaceCoordinate is the value of a single axis of
   -- a point in a workspace.
WorkspaceCoordinate ::= INTEGER (-21845..43690)

-- WorkspaceDeleteReason
   -- This value represents the reason codes for the
   -- WorkspaceDeletePDU.
WorkspaceDeleteReason ::= CHOICE
{
    userInitiated                   NULL,
                                        -- Workspace deletion initiated by user
    insufficientStorage             NULL,
                                        -- Workspace deleted due to insufficient storage capacity.
    nonStandardReason               NonStandardParameter,
    ...
}

-- WorkspaceCreatePDU
   -- This PDU causes a workspace to be created and its
   -- attributes to be set.
WorkspaceCreatePDU  ::= SEQUENCE
{
    workspaceIdentifier             WorkspaceIdentifier,
                                        -- Identifier that will be used to reference this workspace
                                        -- in the future
    appRosterInstance               INTEGER (0..65535),
                                        -- Indicates which application roster instance (returned
                                        -- in the GCC-Application-Roster-Report indication) was
                                        -- valid when this PDU was issued. This is used to eliminate
                                        -- race conditions that can occur when terminals enter a
                                        -- session while a workspace is being created.
    synchronized                    BOOLEAN,
                                        -- TRUE specifies that the workspace contents stacking
                                        -- order must be consistent everywhere.
                                        -- In many cases, this implies the use of
                                        -- MCS-UNIFORM-SEND-DATA for SIPDU submission.
                                        -- FALSE specifies that the workspace contents do not have
                                        -- to be consistent in stacking order; therefore the use of
                                        -- MCS-SEND-DATA is acceptable for all content submitting
                                        -- transactions.
    acceptKeyboardEvents            BOOLEAN,
                                        -- If TRUE this workspace can accept remote
                                        -- keyboard events.
    acceptPointingDeviceEvents      BOOLEAN,
                                        -- If TRUE this workspace can accept remote pointer device events.
    protectedPlaneAccessList        SET (SIZE (1..65536)) OF MCSUserID OPTIONAL,
                                        -- The ability to modify any protected plane in this workspace is
                                        -- restricted only to SICEs on this list. The creator of the workspace
                                        -- is NOT automatically granted access to these planes unless
                                        -- explicitly on this list.
    workspaceSize                   WorkspaceSize,
                                        -- This value specifies the width and height of the new
                                        -- workspace in pixels.
    workspaceAttributes             SET OF WorkspaceAttribute OPTIONAL,
```

```
                                    -- Editable attributes of the workspace
    planeParameters                 SEQUENCE (SIZE (1..256)) OF SEQUENCE
                                    -- This sequence contains plane parameters.
                                    -- Its length is the number
                                    -- of planes in the workspace.
    {
        editable                    BOOLEAN,
                                    -- This item specifies whether objects created on this plane are
                                    -- editable.
                                    -- If not editable, each plane is treated as a bit-map image.
        usage                       SET (SIZE (1..MAX)) OF PlaneUsage,
                                    -- This item specifies restrictions on the usage of this plane
                                    -- (image data or annotation data). At least one use shall be
                                    -- included. A particular usage designator shall be listed no
                                    -- more than once.
        planeAttributes             SET OF PlaneAttribute OPTIONAL,
                                    -- List of attributes
                                    -- A particular attribute shall be listed no more than once.
        ...
    },
    viewParameters                  SET (SIZE (1..256)) OF SEQUENCE
                                    -- Each entry in this list (if any) defines a view to be created in
                                    -- association with this workspace.
    {
        viewHandle                  Handle,
                                    -- Identifier of the view to be created
        viewAttributes              SET OF WorkspaceViewAttribute OPTIONAL,
                                    -- Editable attribute of the view
        ...
    }   OPTIONAL,
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                    -- Allowed only if the corresponding non-standard capabilities
                                    -- are present in the negotiated capability set
    ...,
    refresh                         BOOLEAN OPTIONAL
                                    -- When set to TRUE, the workspace create is being used to
                                    -- refresh a workspace.
                                    -- FALSE or not present indicates that the workspace create
                                    -- represents new session data.
}

-- WorkspaceIdentifier
WorkspaceIdentifier ::= CHOICE
{
    activeWorkspace                 Handle,
                                    -- Handle identifying the active workspace
    archiveWorkspace                SEQUENCE
    {
        archiveHandle               Handle,
                                    -- Handle identifying the archive in which the archived
                                    -- workspace is contained
        entryName                   ArchiveEntryName,
                                    -- Name of the archived workspace
        modificationTime            GeneralizedTime OPTIONAL
                                    -- If the workspace identifier is being used for an operation
                                    -- in which the workspace is to be modified, this parameter
                                    -- shall indicate the time of modification.
                                    -- In this case, the archive header is modified to reflect the
                                    -- most recent modification time. Otherwise, this parameter
                                    -- shall not be included.

    },
    ...
}
```

```
-- WorkspacePoint
   -- A WorkspacePoint is a two-dimensional address of a location in a
   -- workspace plane including points in the invisible border areas.
WorkspacePoint ::= SEQUENCE
{
    xCoordinate         WorkspaceCoordinate,
                            -- X component of a Cartesian address
    yCoordinate         WorkspaceCoordinate
                            -- Y component of a Cartesian address
}

-- WorkspaceRegion
   -- This type can be used to describe both the size and
   -- position of a rectangular region within a workspace.
WorkspaceRegion ::= SEQUENCE
{
    upperLeft           WorkspacePoint,
    lowerRight          WorkspacePoint
}

-- WorkspaceSize
    -- The size of a workspace in pixels.
WorkspaceSize ::= SEQUENCE
{
    width                               INTEGER (1..21845),
                                            -- The number of pixels horizontally
    height                              INTEGER (1..21845)
                                            -- The number of pixels vertically
}

-- WorkspaceViewAttribute
   -- View attributes are editable characteristics of workspace views.
WorkspaceViewAttribute ::= CHOICE
{
    viewRegion                          CHOICE
    {
        fullWorkspace                       NULL,
                                                -- View the entire workspace
        partialWorkspace                    WorkspaceRegion
                                                -- Rectangle defining the region of the workspace to view.
                                                -- The view shall not extend beyond the boundaries of
                                                -- the workspace.
    },
    viewState                           WorkspaceViewState,
                                            -- Visibility state of the view
    updatesEnabled                      BOOLEAN,
                                            -- If this attribute is set to FALSE (the default is TRUE), it
                                            -- is an indication that subsequent updates to the workspace
                                            -- corresponding to this view not be shown until this attribute
                                            -- is set to TRUE.
    sourceDisplayIndicator              SourceDisplayIndicator,
                                            -- Indicates the characteristics of the view within the source
                                            -- display device
    nonStandardAttribute                NonStandardParameter,
    ...
}

-- WorkspaceViewState
   -- A view's state indicates how the local terminal should
   -- display the view.
WorkspaceViewState ::= CHOICE
{
    hidden                              NULL,
                                            -- This workspace should not be shown to the user.
    background                          NULL,
                                            -- The display of this workspace is optional.
    foreground                          NULL,
```

```
                                    -- The display of this workspace is desirable.
    focus                           NULL,
                                    -- The display of this workspace is mandatory. Only one
                                    -- workspace may be set to this state.
    nonStandardState                NonStandardIdentifier,
    ...
}

-- ZOrder
    -- This enumerated type is used to specify a transition to front
    -- or back of an object within an addressable plane.
ZOrder ::= ENUMERATED
{
    front                           (0),
                                    -- Move object to the front of the plane display list
    back                            (1),
                                    -- Move object to the back of the plane display list
    ...
}

-- Begin SIPDU Definitions

-- ArchiveAcknowledgePDU
    -- The ArchiveAcknowledgePDU is used to acknowledge that an archive
    -- has been successfully opened.
ArchiveAcknowledgePDU ::= SEQUENCE
{
    archiveHandle                   Handle,
                                    -- Unique handle that references the archive
    result                          ArchiveOpenResult,
                                    -- Indicates whether or not the archive was opened successfully
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                    -- Allowed only if the corresponding non-standard capabilities
                                    -- are present in the negotiated capability set.
    ...
}

-- ArchiveClosePDU
    -- The ArchiveClosePDU is used to close an archive that was
    -- previously opened during an SI session.
ArchiveClosePDU ::= SEQUENCE
{
    archiveHandle                   Handle,
                                    -- Unique handle that references the archive
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                    -- Allowed only if the corresponding non-standard capabilities
                                    -- are present in the negotiated capability set.
    ...
}

-- ArchiveErrorPDU
    -- The ArchiveErrorPDU is used by a terminal receiving an
    -- archive PDU to signal error conditions to the sender.
ArchiveErrorPDU ::= SEQUENCE
{
    archiveHandle                   Handle,
                                    -- Unique handle that references the archive
    entryName                       ArchiveEntryName OPTIONAL,
                                    -- Specifies the archive entry associated with the error if
                                    -- applicable
    errorCode                       ArchiveError,
                                    -- Specifies the cause of the error at the remote terminal
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                    -- Allowed only if the corresponding non-standard capabilities
                                    -- are present in the negotiated capability set.
    ...
```

```
}

-- ArchiveOpenPDU
   -- The ArchiveOpenPDU is used to open an archive at a
   -- remote terminal that supports this capability.
ArchiveOpenPDU ::= SEQUENCE
{
    archiveHandle                   Handle,
                                        -- Unique handle that is used to reference this archive during
                                        --  the session
    mode                            ArchiveMode,
                                        -- Indicates the access restrictions placed on the archive
    header                          ArchiveHeader,
                                        -- Specifies information used to identify the archive. If the archive
                                        -- is being created, this is the information that is used to identify
                                        -- the archive in the future.
    maxEntries                      INTEGER (1..65535) OPTIONAL,
                                        -- This parameter allows remote terminals to estimate the local
                                        -- resource usage for the specified archive so they can signal an
                                        -- error early in the archiving process. It is only to be specified if
                                        -- the archive open mode is set to "create".
    nonStandardParameters            SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities are
                                        -- present in the negotiated capability set.
    ...
}

-- BitmapAbortPDU
   -- This PDU is used by both the transmitting SICE to signal that a
   -- bitmap exchange is being aborted and by an SICE requesting that
   -- a bitmap exchange in progress be aborted.
BitmapAbortPDU ::= SEQUENCE
{
    bitmapHandle                    Handle,
                                        -- Handle referring to the bitmap being created
    userID                          MCSUserID OPTIONAL,
                                        -- Optionally provided by the transmitter if identification of
                                        -- the source of the abort is desired
    reason                          BitmapAbortReason OPTIONAL,
    message                         BMPString (SIZE (1..256)) OPTIONAL,
    nonStandardParameters            SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}
-- BitmapCheckpointPDU
   -- This PDU is used by a terminal that is transmitting a bitmap
   -- when it wants to notify receiving terminals that they should
   -- display previously received data.
BitmapCheckpointPDU ::= SEQUENCE
{
    bitmapHandle                    Handle,
                                        -- Handle used to reference this bitmap
    passedCheckpoints               SET (SIZE (1..100)) OF TokenID,
                                        -- List of checkpoints that have been uninhibited by all nodes
    percentComplete                 INTEGER (1..100),
                                        -- Cumulative portion of the bitmap completed as a result of
                                        -- all passed checkpoints so far
    nonStandardParameters            SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- BitmapCreatePDU
   -- This PDU is used to initiate a bitmap transmission.
```

```
BitmapCreatePDU ::= SEQUENCE
{
    bitmapHandle                    Handle,
                                        -- Handle to be used to reference this object in the future
    destinationAddress              BitmapDestinationAddress,
                                        -- Destination address of the bitmap
    attributes                      SET OF BitmapAttribute OPTIONAL,
                                        -- List of editable attributes of the bitmap
    anchorPoint                     WorkspacePoint OPTIONAL,
                                        -- Point of origin of the bitmap with respect to the
                                        -- destination workspace. Only needed for softcopy bitmaps.
                                        -- Default is (0,0).
    bitmapSize                      BitmapSize,
                                        -- Width and height of the total bitmap represented in the
                                        -- bitstream. For a multi-component bitmap, this is the
                                        -- size of the largest component.
    bitmapRegionOfInterest          BitmapRegion OPTIONAL,
                                        -- Region of interest within the bitmap to be applied to the
                                        -- workspace
                                        -- Default is full bitmap.
    pixelAspectRatio                PixelAspectRatio,
                                        -- Pixel aspect ratio of the bitmap
    scaling                         PointDiff16 OPTIONAL,
                                        -- Offset in workspace coordinates of the lower right hand
                                        -- corner of the bitmap relative to the anchor point.
                                        -- Default is no scaling.
                                        -- Only needed for softcopy bitmaps.
    checkpoints                     SEQUENCE (SIZE (1..100)) OF TokenID OPTIONAL,
                                        -- Tokens to be used for checkpointing the bitmap create
                                        --  exchange
    bitmapFormatHeader              CHOICE
                                        -- The following headers provide image bitstream parameters
                                        -- that are outside the scope of the corresponding coding
                                        -- standard but are necessary for image decompression.
                                        -- NOTE – Some bitmap formats are disallowed depending
                                        -- on the value of the destinationAddress parameter.
    {
        bitmapHeaderUncompressed        BitmapHeaderUncompressed,
                                            -- Parameters for the uncompressed pixel representation
        bitmapHeaderT4                  BitmapHeaderT4,
                                            -- Parameters for T4 (G3) encoded bitstreams outside
                                            -- the T.4 standard's scope
        bitmapHeaderT6                  BitmapHeaderT6,
                                            -- Parameters for T6 (G4) encoded bitstreams outside
                                            -- the T.6 standard's scope
        bitmapHeaderT81                 BitmapHeaderT81,
                                            -- Parameters for T.81 (JPEG) encoded bitstreams
                                            -- outside the T.81 standard's scope
        bitmapHeaderT82                 BitmapHeaderT82,
                                            -- Parameters for T.82 (JBIG) encoded bitstreams
                                            -- outside the T.82 standard's scope
        bitmapHeaderNonStandard         NonStandardParameter,
        ...
    },
    bitmapData                      BitmapData OPTIONAL,
                                        -- Compression format specific bitmap data padded to
                                        --  be byte-aligned.
    moreToFollow                    BOOLEAN,
                                        -- Indicates whether or not this is the last block of data for
                                        -- the bitmap
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}
```

```
-- BitmapCreateContinuePDU
   -- This PDU is used by the transmitting SICE to continue
   -- a bitmap transmission begun by a BitmapCreatePDU
BitmapCreateContinuePDU ::= SEQUENCE
{
    bitmapHandle                    Handle,
                                        -- Handle referring to the bitmap being created
    bitmapData                      BitmapData,
                                        -- Bitmap data
    moreToFollow                    BOOLEAN,
                                        -- Indicates whether or not this is the last block of data for
                                        -- the bitmap
    nonStandardParameters             SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- BitmapDeletePDU
   -- This PDU is used to delete bitmaps.
BitmapDeletePDU ::= SEQUENCE
{
    bitmapHandle                    Handle,
                                        -- Handle used to reference this bitmap
    nonStandardParameters             SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- BitmapEditPDU
    -- This PDU is used to change bitmap attributes.
BitmapEditPDU ::= SEQUENCE
{
    bitmapHandle                    Handle,
                                        -- Handle used to reference this bitmap
    attributeEdits                  SET OF BitmapAttribute OPTIONAL,
                                        -- List of attributes to be edited
    anchorPointEdit                 WorkspacePoint OPTIONAL,
                                        -- Point of origin of the bitmap with respect to the destination
                                        -- workspace
    bitmapRegionOfInterestEdit      BitmapRegion OPTIONAL,
                                        -- Region of interest within the bitmap to be applied to the
                                        -- workspace
    scalingEdit                     PointDiff16 OPTIONAL,
                                        -- Offset in workspace coordinates of the lower right hand
                                        -- corner of the bitmap relative to the anchor point
    nonStandardParameters             SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- ConductorPrivilegeGrantPDU
   -- This PDU is used by the conductor to grant or revoke privileges
   -- when the session is in conducted mode.
ConductorPrivilegeGrantPDU ::= SEQUENCE
{
    destinationUserID               MCSUserID,
                                        -- MCS User ID of the destination node
    privilegeList                   SET OF ConductorPrivilege,
                                        -- A particular privilege shall appear in this list no more
                                        -- than once.
    nonStandardParameters             SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
```

```
    ...
}

-- ConductorPrivilegeRequestPDU
   -- This PDU is used to request privileges from the conductor
   -- when the session is in conducted mode.
ConductorPrivilegeRequestPDU ::= SEQUENCE
{
    privilegeList                   SET OF ConductorPrivilege,
                                        -- A particular privilege shall appear in this list no more
                                        -- than once.
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- DrawingCreatePDU
   -- A drawingCreate PDU is used to deposit one or more
   -- drawing elements to a workspace plane.
DrawingCreatePDU ::= SEQUENCE
{
    drawingHandle                   Handle OPTIONAL,
                                        -- Handle to be used to reference this drawing object in
                                        -- future exchanges. Note that editing and deleting objects is
                                        -- only valid if the target plane is of type "editable".
    destinationAddress              DrawingDestinationAddress,
                                        -- Destination of drawing
    drawingType                     DrawingType,
                                        -- Which basic drawing shape this element represents
    attributes                      SET OF DrawingAttribute OPTIONAL,
                                        -- Attributes of the drawing object.
                                        -- NOTE – All attributes have default values that are assumed
                                        -- if the attribute is not specified.
    anchorPoint                     WorkspacePoint,
                                        -- Point of origin of the drawing element.
                                        -- This forms the first of the control points and is the point
                                        -- from which all other control points are defined relative to.
    rotation                        RotationSpecifier OPTIONAL,
                                        -- Specifies a rotation angle and point of revolution for
                                        -- the drawing element
    sampleRate                      INTEGER (1..255) OPTIONAL,
                                        -- For applicable types, this indicates the rate at which
                                        -- points were acquired by the transmitting terminal
                                        -- (in samples per second) so they can be replayed at
                                        -- a similar rate if desired
    pointList                       PointList,
                                        -- List of control points that define the drawing shape.
                                        -- The interpretation of the control point list is dependent on
                                        -- the value of the "type" parameter.
                                        -- Note that the control points in the list are differentially
                                        -- encoded from the previous.
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- DrawingDeletePDU
   -- A DrawingDeletePDU is used to delete one
   -- graphical element from a workspace plane.
DrawingDeletePDU ::= SEQUENCE
{
    drawingHandle                   Handle,
                                        -- Drawing object to delete
```

```
    nonStandardParameters               SET OF NonStandardParameter OPTIONAL,
                                            -- Allowed only if the corresponding non-standard capabilities
                                            -- are present in the negotiated capability set.
    ...
}

-- DrawingEditPDU
   -- A DrawingEditPDU is used to alter one or more of
   -- a drawing element's attributes or parameters.
DrawingEditPDU ::= SEQUENCE
{
    drawingHandle                       Handle,
                                            -- Identifier of item to be edited
    attributeEdits                      SET OF DrawingAttribute OPTIONAL,
                                            -- List of attribute changes
    anchorPointEdit                     WorkspacePoint OPTIONAL,
                                            -- Point of origin of the drawing element
    rotationEdit                        RotationSpecifier OPTIONAL,
                                            -- Specifies a rotation angle and point of revolution for the
                                            -- drawing element
    pointListEdits                      PointListEdits OPTIONAL,
                                            -- List of control point changes.
                                            -- Note that the index refers to
                                            -- the point list not including the anchor point.
    nonStandardParameters               SET OF NonStandardParameter OPTIONAL,
                                            -- Allowed only if the corresponding non-standard capabilities
                                            -- are present in the negotiated capability set.
    ...
}

-- FontPDU
FontPDU ::= SEQUENCE
{
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                            -- Allowed only if the corresponding non-standard
                                            -- capabilities are present in the negotiated capability set.
    ...
}

-- RemoteEventPermissionGrantPDU
   -- This PDU is used to grant permission to issue remote
   -- events.
RemoteEventPermissionGrantPDU ::=  SEQUENCE
{
    destinationAddress                  RemoteEventDestinationAddress,
                                            -- Address to which remote event permission is being granted
    destinationUserID                   MCSUserID,
                                            -- MCS User ID of the destination node
    remoteEventPermissionList           SET OF RemoteEventPermission,
                                            -- A particular permission shall not be included in this list more
                                            -- than once.
    nonStandardParameters               SET OF NonStandardParameter OPTIONAL,
                                            -- Allowed only if the corresponding non-standard capabilities are
                                            -- present in the negotiated capability set.
    ...
}

-- RemoteEventPermissionRequestPDU
   -- This PDU is used to request permission to issue remote
   -- events from the workspace creator.
RemoteEventPermissionRequestPDU ::=  SEQUENCE
{
    destinationAddress                  RemoteEventDestinationAddress,
                                            -- Address to which remote event permission is being requested
```

```
    remoteEventPermissionList       SET OF RemoteEventPermission,
                                        -- A particular permission shall not be included in this list more
                                        -- than once.
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- RemoteKeyboardEventPDU
   -- This PDU signals keyboard event.
RemoteKeyboardEventPDU ::= SEQUENCE
{
    destinationAddress              RemoteEventDestinationAddress,
                                        -- Destination address of remote event
    keyModifierStates               SET OF KeyModifier OPTIONAL,
                                        -- Set of key modifiers. Only modifiers in this list
                                        -- are assumed to be active.
                                        -- A particular key modifier shall not be included
                                        -- in this set more than once.
    keyPressState                   KeyPressState,
                                        -- This item specifies keyboard event that is being signaled
    keyCode                         KeyCode,
                                        -- Character corresponding to the pressed key or function key
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- RemotePointingDeviceEventPDU
   -- This PDU is used to signal pointing device events.
RemotePointingDeviceEventPDU ::= SEQUENCE
{
    destinationAddress              RemoteEventDestinationAddress,
                                        -- Destination address of remote event
    leftButtonState                 ButtonEvent,
                                        -- This specifies the left button state.
    middleButtonState               ButtonEvent,
                                        -- This specifies the middle button state.
    rightButtonState                ButtonEvent,
                                        -- This specifies the right button state.
    initialPoint                    WorkspacePoint,
                                        -- This specifies the initial pointing device position.
    sampleRate                      INTEGER (1..255) OPTIONAL,
                                        -- This parameter indicates the rate at which points were
                                        -- acquired by the transmitting terminal (in samples per second)
                                        -- so they can be replayed at a similar rate if desired.
    pointList                       PointList OPTIONAL,
                                        -- Additional coordinates that are each differentially encoded
                                        -- with respect to the initialPoint parameter
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- RemotePrintPDU
   -- This PDU is used to instruct a remote terminal to print
   -- the specified workspace.
RemotePrintPDU ::= SEQUENCE
{
    destinationAddress              RemoteEventDestinationAddress,
                                        -- Destination address of remote event
    numberOfCopies                  INTEGER (1..65536) OPTIONAL,
                                        -- Number of copies to be printed
```

```
    portrait                        BOOLEAN OPTIONAL,
                                        -- TRUE specifies a portrait paper orientation.
                                        -- FALSE specifies a landscape paper orientation.
    regionOfInterest                WorkspaceRegion OPTIONAL,
                                        -- Optionally defines rectangular region of interest
                                        -- within the workspace to be printed.
                                        -- If not present, it is implied that
                                        -- the entire workspace is to be printed.
    nonStandardParameters            SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- SINonStandardPDU
   -- This PDU allows any non-standard information to be transmitted.
SINonStandardPDU ::= SEQUENCE
{
    nonStandardTransaction          NonStandardParameter,
    ...
}

-- SIPDU
   -- The set of all SIPDUs.
SIPDU ::= CHOICE
{
    archiveAcknowledgePDU           ArchiveAcknowledgePDU,
    archiveClosePDU                 ArchiveClosePDU,
    archiveErrorPDU                 ArchiveErrorPDU,
    archiveOpenPDU                  ArchiveOpenPDU,

    bitmapAbortPDU                  BitmapAbortPDU,
    bitmapCheckpointPDU             BitmapCheckpointPDU,
    bitmapCreatePDU                 BitmapCreatePDU,
    bitmapCreateContinuePDU         BitmapCreateContinuePDU,
    bitmapDeletePDU                 BitmapDeletePDU,
    bitmapEditPDU                   BitmapEditPDU,

    conductorPrivilegeGrantPDU      ConductorPrivilegeGrantPDU,
    conductorPrivilegeRequestPDU    ConductorPrivilegeRequestPDU,

    drawingCreatePDU                DrawingCreatePDU,
    drawingDeletePDU                DrawingDeletePDU,
    drawingEditPDU                  DrawingEditPDU,

    remoteEventPermissionGrantPDU   RemoteEventPermissionGrantPDU,
    remoteEventPermissionRequestPDU RemoteEventPermissionRequestPDU,
    remoteKeyboardEventPDU          RemoteKeyboardEventPDU,
    remotePointingDeviceEventPDU    RemotePointingDeviceEventPDU,
    remotePrintPDU                  RemotePrintPDU,

    siNonStandardPDU                SINonStandardPDU,

    workspaceCreatePDU              WorkspaceCreatePDU,
    workspaceCreateAcknowledgePDU   WorkspaceCreateAcknowledgePDU,
    workspaceDeletePDU              WorkspaceDeletePDU,
    workspaceEditPDU                WorkspaceEditPDU,
    workspacePlaneCopyPDU           WorkspacePlaneCopyPDU,
    workspaceReadyPDU               WorkspaceReadyPDU,
    workspaceRefreshStatusPDU       WorkspaceRefreshStatusPDU,
    ...,

    -- PDUs Added During 1st Revision
    fontPDU                         FontPDU,

    textCreatePDU                   TextCreatePDU,
    textDeletePDU                   TextDeletePDU,
    textEditPDU                     TextEditPDU,
```

```
    videoWindowCreatePDU            VideoWindowCreatePDU,
    videoWindowDeletePDU            VideoWindowDeletePDU,
    videoWindowEditPDU              VideoWindowEditPDU
}

-- TextCreatePDU
TextCreatePDU ::= SEQUENCE
{
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- TextDeletePDU
TextDeletePDU ::= SEQUENCE
{
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- TextEditPDU
TextEditPDU ::= SEQUENCE
{
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- VideoWindowDeletePDU
   -- This PDU deletes video windows.
VideoWindowDeletePDU ::= SEQUENCE
{
    videoWindowHandle               Handle,
                                        -- Handle referencing the video window
                                        -- to be deleted.
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
   ...
}

-- WorkspaceCreateAcknowledgePDU
   -- This PDU acknowledges the reception of a WorkspaceCreatePDU in
   -- the case of unsynchronized workspace.
WorkspaceCreateAcknowledgePDU ::= SEQUENCE
{
    workspaceIdentifier             WorkspaceIdentifier,
                                        -- Workspace being acknowledged.
    nonStandardParameters           SET OF NonStandardParameter OPTIONAL,
                                        -- Allowed only if the corresponding non-standard capabilities
                                        -- are present in the negotiated capability set.
    ...
}

-- WorkspaceDeletePDU
   -- This PDU causes a workspace to be destroyed.
WorkspaceDeletePDU ::= SEQUENCE
{
    workspaceIdentifier             WorkspaceIdentifier,
                                        -- Workspace to be deleted
    reason                          WorkspaceDeleteReason,
                                        -- Reason for deletion of the workspace
```

```
    nonStandardParameters               SET OF NonStandardParameter OPTIONAL,
                                            -- Allowed only if the corresponding non-standard capabilities
                                            -- are present in the negotiated capability set.
    ...
}

-- WorkspaceEditPDU
   -- This PDU allows workspace attributes to be edited.
WorkspaceEditPDU ::= SEQUENCE
{
    workspaceIdentifier                 WorkspaceIdentifier,
                                            -- Workspace to be edited
    attributeEdits                      SET OF WorkspaceAttribute OPTIONAL,
                                            -- List of attribute changes

    planeEdits                          SET (SIZE (1..256)) OF SEQUENCE
    {
        plane                           DataPlaneID,
                                            -- Plane whose attributes are to be edited
        planeAttributes                 SET OF PlaneAttribute,
                                            -- List of attributes to change
        ...
    } OPTIONAL,

    viewEdits                           SET (SIZE (1..256)) OF SEQUENCE
    {
        viewHandle                      Handle,
                                            -- Identifier of the view to be edited
        action                          CHOICE
        {
            createNewView               SET OF WorkspaceViewAttribute,
                                                -- Editable attributes of the view
            editView                    SET OF WorkspaceViewAttribute,
                                                -- List of attributes to change
            deleteView                  NULL,
            nonStandardAction           NonStandardParameter,
            ...
        },
        ...
    } OPTIONAL,

    nonStandardParameters               SET OF NonStandardParameter OPTIONAL,
                                            -- Allowed only if the corresponding non-standard
                                            -- capabilities are present in the negotiated capability set.
    ...
}

-- WorkspacePlaneCopyPDU
   -- This PDU causes a portion of a plane to be copied
   -- to another plane (either intra- or inter-workspace).
   --  The source and destinations must either both be
   -- permanent or both be editable, and they must have the same
   -- usage designator; otherwise copy for that plane will
   -- not take place.
   -- If the planes are editable, objects with any of their control
   -- points falling totally within the source rectangle are copied.
   -- If the Scaling capability has been negotiated in the case of a
   -- softcopy workspace then it is not necessary for the source and
   -- destination rectangles to be the same size.
WorkspacePlaneCopyPDU ::= SEQUENCE
{
    sourceWorkspaceIdentifier           WorkspaceIdentifier,
                                            -- Workspace to be copied
    sourcePlane                         DataPlaneID,
                                            -- Source plane identifier
```

```
    destinationWorkspaceIdentifier      WorkspaceIdentifier,
                                            -- Destination workspace identifier.
                                            -- May be the same as the source workspace
    destinationPlane                    DataPlaneID,
                                            -- Destination plane identifier.
                                            -- May be the same as the source plane
    copyDescriptor                      CHOICE
    {
        permanentPlaneCopyDescriptor        PermanentPlaneCopyDescriptor,
        editablePlaneCopyDescriptor         EditablePlaneCopyDescriptor,
        ...
    },
    nonStandardParameters                SET OF NonStandardParameter OPTIONAL,
                                            -- Allowed only if the corresponding non-standard capabilities
                                            -- are present in the negotiated capability set.
    ...
}

-- WorkspaceReadyPDU
   -- This PDU signals that a workspace create is complete (for
   -- unsynchronized workspaces).
WorkspaceReadyPDU ::= SEQUENCE
{
    workspaceIdentifier                 WorkspaceIdentifier,
                                            -- Workspace being enabled
    nonStandardParameters                SET OF NonStandardParameter OPTIONAL,
                                            -- Allowed only if the corresponding non-standard capabilities
                                            -- are present in the negotiated capability set.
    ...
}

-- WorkspaceRefreshStatusPDU
   -- This PDU is used by an SICE to announce or remit its status as
   -- the session refresh SICE for SICEs that join late.
WorkspaceRefreshStatusPDU ::= SEQUENCE
{
    refreshStatus                       BOOLEAN,
                                            -- TRUE indicates that the SICE sourcing this PDU is
                                            -- functioning as the session-wide refresher.
                                            -- FALSE indicates that the SICE sourcing this PDU has
                                            -- ceased to function as the session-wide refresher.
    nonStandardParameters               SET OF NonStandardParameter OPTIONAL,
                                            -- Allowed only if the corresponding non-standard capabilities
                                            -- are present in the negotiated capability set.
    ...
}

-- End SI Definitions
END
```

Annex A

SI profiles

(This annex forms an integral part of this Recommendation)

The following profiles are defined as guidelines for terminal vendors wishing to build equipment that is maximally interoperable. Note that the protocol itself explicitly does not support a profile shorthand mechanism and requires explicit advertizement of each capability. This is to insure proper forward compatibility. A terminal conforms to a profile if it advertizes capabilities greater than or equal to the minimums specified.

Table A.1 – SI profiles

Capability name	Hard-Copy-0	Soft-Copy-Image-0	Soft-Copy-Image-1	Soft-Copy-White-Board-0	Soft-Copy-Annotated-Image-0
Hard-Copy-Image	M	O	O	O	O
Soft-Copy-Workspace	O	M	M	M	M
Soft-Copy-Workspace-Max-Width	O	O	≥ 768	O	≥ 768
Soft-Copy-Workspace-Max-Height	O	O	≥ 576	O	≥ 576
Soft-Copy-Workspace-Max-Planes	O	O	O	O	≥ 2
Soft-Copy-Pointing	O	O	O	O	M
Soft-Copy-Annotation	O	O	O	M	M
Soft-Copy-Image	O	M	M	O	M
Soft-Copy-Image-Bitmap-Max-Width	O	O	≥ 768	O	≥ 768
Soft-Copy-Image-Bitmap-Max-Height	O	O	≥ 576	O	≥ 576

The definition of each of these capabilities is listed in Table 8-1. The requirements for specific bitmap encoding algorithms with respect to these capabilities are included in clauses 8.5.6.1 (uncompressed encoding) 8.5.6.2 (T.4 encoding), 8.5.6.4 (T.81 encoding) and 8.5.6.5 (T.82 encoding).

NOTE – Workspace resolutions are measured with respect to a square pixel format but the actual pixel format of exchanged bitmaps can be variable.

Any capabilities which are not listed in Table A.1 are optional for all profiles. Note that some capabilities depend on the presence of others. It is a protocol violation to include any dependent capability in the application capabilities list if the capability on which it depends is not also included. These dependencies are listed in Table 8-1.

Annex B

Object identifier assignments

(This annex forms an integral part of this Recommendation)

Table B.1 lists the assignment of object identifiers defined for use by this Recommendation.

Table B.1 – Object identifiers defined in T.126

Object identifier value	Description
{itu-t recommendation t 126 version(0) 2}	This object identifier is used to indicate the version of this Recommendation. At this time there is a single standardized version defined.

Appendix I

Deriving intermediate palettes for bitplane progressive transmission of palettized images

(This appendix does not form an integral part of this Recommendation)

It is possible to derive a set of colour index tables, one per bitplane sent, that form interim palettes by referencing colours in the final bitmap palette that provide acceptable mappings for each intermediate image built from all the bitplanes transmitted, up to and including the most current. The derivation approach is based on a Kd-tree vector quantizer, which allows binary subdivisions of the colour space defined by the original image palette.

One candidate algorithm follows but many variants are possible. These inventions are left to the discretion of the implementer.

Example palette splitting algorithm:

– Initialize the root node of a binary tree to contain all the colour entries in the original image palette.

– Find a single suitable representative colour for this node from this set. Note that the representative colour must be in the original palette. Suitable representatives can be found using techniques such as determining the average value over the set, etc., followed by a matching pass over the bitmap palette to find the closest match.

For each bitplane

– For each leaf

  • Create two children for the current leaf node, each inheriting one part of the constituent palette entries from its parent node. Each colour palette entry must be uniquely assigned to one child or the other. This is done by determining median value point along the axis with the maximum error with respect to the selected representative colour, and distributing all the constituents that are less than or equal to that value along the split axis to the left child and all those greater than the split point to the right child.

  • Calculate a suitable representative colour for each node from the nodes' new colour set. Note that the representative colour must be in the original palette. Suitable representatives can be found using techniques such as determining the average value over the set, etc. followed by a matching pass over the bitmap palette to find the closest match.

– End for each leaf

  • Walk the tree and extract the representative colour from each leaf. Each of these leaf colours are then converted to indices by finding the position in the bitmap palette containing the closest matching colour to the leaf representative colour. Each of these gets placed in the colour index table position whose corresponding address has a prefix that matches the string of zeros and ones that describes the path from the root of the tree to that leaf. Zero indicates the taking of a left branch, one indicates a right branch. The string must be populated from MSB to LSB forming the address prefix of the colour index table to be filled with the index value referencing the bitmap palette containing the closest match to that leaf representative colour.

  • Store the colour index table formed in the above step to be used for the image formed by bit planes up to and including the bit plane corresponding to the current depth level of the tree.

– End for each bitplane

NOTE – The final bitplane colour index table can be omitted if the bitmap palette is reordered locally by permuting it as specified by the colour index table derived for the last bitplane. This new bitmap palette would become the bitmap palette for the image. If this is done, all other colour index table values for previous bitplanes must be changed to reference the new reordered bitmap palette. Also note that the pixel data must be reordered by the transmitter before coding, such that it maps properly to either the final bitplane's colour index table or the bitmap palette depending on which of the previously mentioned approaches is chosen.

Figure I.1 illustrates how the treewise palette splitting will ultimately look after all split iterations are complete.

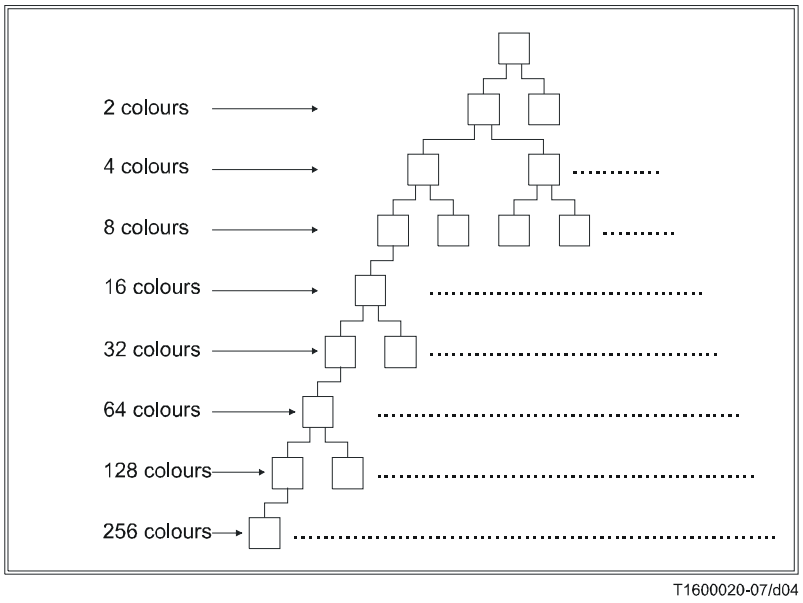

Figure I.1 – Palette splitting

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	**Terminals for telematic services**
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	Global information infrastructure, Internet protocol aspects and next-generation networks
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2008