

REIHE INFORMATIK

5/2001

RTP/I Payload Type Definition for Shared Whiteboards

Jürgen Vogel

Universität Mannheim

Praktische Informatik IV

L15, 16

D-68131 Mannheim

RTP/I Payload Type Definition for Shared Whiteboards

Jürgen Vogel
Department for Praktische Informatik IV
University of Mannheim
vogel@informatik.uni-mannheim.de

Abstract

This document specifies an application-level protocol (i.e., payload type) for interactive shared whiteboards using the Real Time Protocol for Distributed Interactive Media (RTP/I). RTP/I defines a standardized framing for the transmission of application data and provides protocol mechanisms that are universally needed for the class of distributed interactive media. A shared whiteboard is a tool to present and edit documents (slides) in a collaborative environment (e.g., a video conference). Shared whiteboards belong to the distributed interactive media class and are therefore well suited to base their network communication model on RTP/I. This document shows how to employ RTP/I with shared whiteboards and defines application data units (ADUs) for operations needed by a whiteboard. This protocol definition allows standardized collaboration between different whiteboard implementations.

1 Introduction

Distributed interactive media are media which allow a set of spatially separated users to synchronously interact with the medium itself. Typical examples of distributed interactive media are shared whiteboards which are used to present and edit slides in a teleconferencing environment [Tun98, Gey99], as well as distributed virtual environments (DVEs) [Hag96], shared text editors [HC97], and computer games with network support [GD98]. In the following, we briefly introduce the class of distributed interactive media and the Real Time Protocol for Distributed Interactive Media (RTP/I). For a more detailed discussion please refer to [Mau00, MHKE01, MHK⁺00].

In order to provide high responsiveness and to avoid the drawbacks of centralized approaches, such as the presence of a single point-of-failure and lack of scalability, applications for distributed interactive media often employ a replicated distribution architecture. In this architecture each user runs an instance of the application which manages a local copy of the medium's shared state. For example, the state of a shared whiteboard presentation includes a number of presentation slides, each holding graphical objects such as images and text.

The state of a distributed medium changes either by the passage of time or by means of user interactions (*events*). State changes due to the passage of time can be calculated locally and need not to be distributed among application instances. In contrast, events have to be exchanged among instances via the network to all remote instances of the application so that each can modify its local copy of the state accordingly. User interactions that are issued in a sequence in which one action overwrites the preceding one are called *cues* (e.g., mouse movements). Only the last action of such a sequence is an event. For better handling, the application's state can be partitioned into several *sub-components*.

In some cases, it is desirable that the initial state (i.e., the state at the beginning of a session) of a distributed interactive medium is not empty. For example, it might be useful to distribute a set of presentation slides asynchronously before starting the actual session in order to save bandwidth during the session. The initial state of a medium is called *environment*. Typically, this part of the medium's state is static and does not change during a session. Distribution of the environment is outside the scope of RTP/I and can be done, e.g., by ftp.

RTP/I is an application-level protocol that employs the media model described above, and is applicable to arbitrary distributed interactive media. It consists of two main parts; both reside on the application level and are independent of the underlying network and transport layers:

- the framing protocol (RTP/I). RTP/I is used to frame the data transmitted by distributed interactive media. The RTP/I framing contains the information that is common to media of a specific class. This information makes it possible to understand to a large extent the semantics of the transmitted data without any medium-specific knowledge. Therefore, meaningful functionality and services that are independent of the media-specific data encapsulated by the framing information can be developed.
- the RTP/I control protocol (RTCP/I). RTCP/I is used to convey meta information about the medium and information about the participants in a session.

RTP/I is not a complete protocol. It needs to be adapted to the requirements of a specific medium or a group of media by defining either a payload or a profile. A profile adapts RTP/I to the needs of a group of distributed interactive media. A payload type definition is a specification document that defines how a particular medium is transported using the framework provided by RTP/I. Essentially, it describes how the medium specific-data are encoded and specifies a payload for interactive shared whiteboards. The aim of such a standardized protocol is to allow communication between different whiteboard implementations. The specific encoding "whiteboard" as defined by this document is assigned the payload type "1". Each RTP/I ADU carries this payload type as the identifier of the originating application.

The remainder of this document is structured as follows. First, we explain how RTP/I can be used by a shared whiteboard. Then we define all the necessary application data units (ADUs) of a whiteboard protocol. These ADUs are transported either as RTP/I state, event, or cue. The document concludes with a summarizing list of constants used to build the ADUs.

2 Usage of RTP/I

The state of a shared whiteboard consists of document pages (presentation slides). Each page can hold an arbitrary number of graphical objects such as rectangles, images, and freehand lines. Besides having own attributes, a page is therefore a container for graphical objects. Pages can be organized through folders, i.e., folders contain pages and other (sub-)folders. Graphical objects on a page can form groups. Summing up, whiteboard objects form a tree-like hierarchy. Root of this tree is the session itself; it contains all folders and pages.

All these objects have a state that can be changed by user interaction. Each object is represented by an RTP/I sub-component. For identification, each sub-component is assigned a unique ID. Assignment of IDs is done by a unique number service [MHK⁺00]. The ID "1" is reserved for the root of the hierarchy tree.

Hierarchy information concerning the arrangement of sub-components in the tree are encoded via RTP/I, since sub-component IDs are not allowed as payload of RTP/I packets [MHK⁺00]. In contrast, information about the ordering of sub-components within the same parent container is included in certain RTP/I ADUs: states carry in their header the ID of the sub-components direct parent and in their payload (i.e., in the whiteboard ADU) the ordering index that defines the sub-components place in the sub-component list of its parent. A special event (change parent) indicates changes in the hierarchy. This event contains the ID of the new parent and the ordering index. Indices generally start with "0".

Each user interaction that changes the state of the shared whiteboard (e.g., creating a rectangle, moving an arrow, and changing the color of a line) is translated to an ADU that is distributed via RTP/I as state, event or cue. Creating an object causes transmission of an state. Changing attributes (e.g., color, size, position) of an object or deleting an object causes transmission of events or cues. It is recommended to transmit all temporary state changes as cues and only final state changes as events. For example, when moving an object only the final position should be transmitted as event, and all intermediate positions as cues. The percentage of mouse movements triggering cue transmissions can be chosen individually, depending on available bandwidth and user preferences.

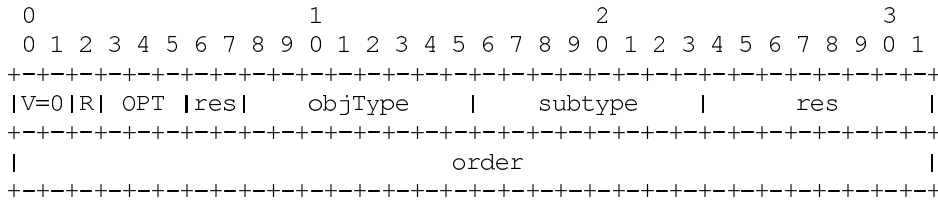
In some cases it might be necessary to transmit the current state of a sub-component (e.g, resynchronization in case of an inconsistency). These states are encoded as RTP/I states (like states used for creating an object). RTP/I events are distinguished into delete and change operations. Cues transport only change operations.

All the sub-components needed to display the medium's state in the shared workspace are marked as active [MHK⁺00]. Typically, there is one active page (including a number of assigned graphical objects) at a certain point in time. Being active (or not) is an important part of a page's state. Changing an active page triggers the transmission of one or more events: if there is only one active page at a certain point in time, one event changes the state of the formerly active page to inactive, and one event changes the state of the page to be displayed now to active. When the whiteboard is able to present more than one page simultaneously, more than pages might be affected.

This document does not specify a certain transport protocol. Rather, it is assumed that the application makes use of a reliable transport mechanism that guarantees the reliable distribution of operations. This mechanism can be integrated either into the application, or into RTP/I, or can be implemented at the transport level.

3 ADU Header

All whiteboard ADUs share a common header (defined below). Therefore it is possible to decide on the correct parsing function on basis of operation type, object type, and subtype. In the following, all fields reserved for future use are marked as "res" or "r".



version (V) : 2 bits

This field specifies the version number of the whiteboard protocol. The version defined by this specification is version 0.

environment reference (R) : 1 bit

This bit is set, if the state of the object is not encoded in the ADU itself but in the environment. The ADU contains an environment reference instead of the actual state. This technique can be used only if the state of the object encoded in the environment is identical to the object's current state.

operation type (OPT) : 3 bits

This field identifies the operation type of the ADU. There are three possible values: 0 for state ADUs, 1 for change ADUs, and 2 for delete ADUs. ADUs with OPT 0 are send as RTP/I states, ADUs with OPTs 1 are send as RTP/I events or cues, and ADUs with OPTs 2 are send as RTP/I events.

object type (objType) : 8 bits

This field identifies the type of the object affected by the ADU. Currently, there are 11 possible values: 0 for session, 1 for folder, 2 for page, 4 for rectangle, 5 for oval, 6 for line, 7 for polyline, 8 for polygon, 9 for text, 10 for image, and 11 for group.

subtype : 8 bits

This field carries further information to classify the operation to be executed.

order : 32 bits

This field defines the ordering of objects with the same direct parent in the hierachy tree. The sorting is ascending.

4 ADUs

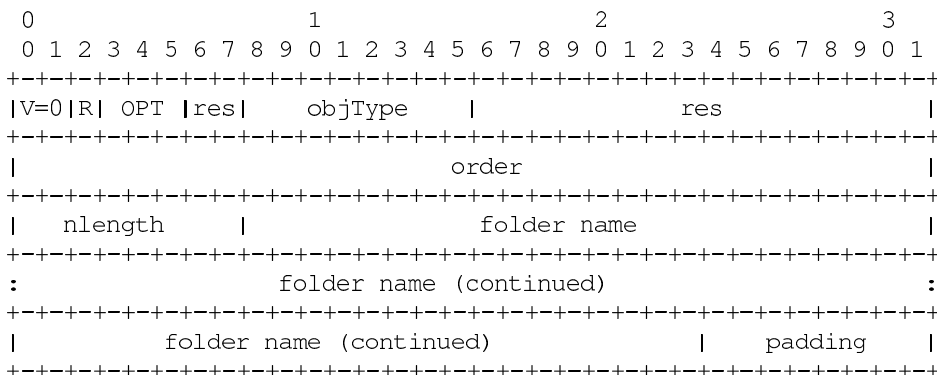
This section gives a complete specification of whiteboard ADUs that are needed to communicate whiteboard functionality. Missing specifications of a certain operation mean that the ADU differs from a preceding specification only by object type or operation type, and a complete listing is therefore not necessary. The same holds true for missing specifications of fields.

4.1 Container ADUs

4.1.1 Folder ADUs

Folders (or chapters) are containers and can hold other folders and pages. Folders have the object type (objType) 1.

4.1.1.1 Folder State ADU



operation type (OPT) : 3 bits
 This field identifies the type of the ADU. It is set to 0 (state).

reserved (res) : 2 bits
 Unused bits.

object type (objType) : 8 bits
 This field identifies the type of the object affected by the ADU.
 It is set to 1 (folder).

name length (nlength) : 8 bits
 This field identifies the length of the folder name in bytes excluding any padding bytes. The maximum name length is 255.

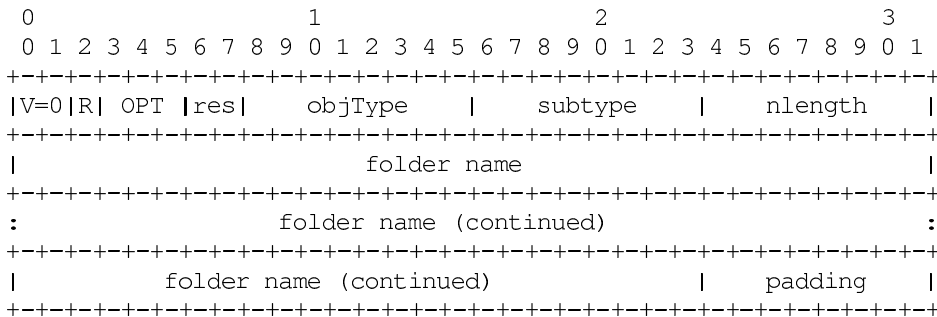
folder name : length as specified in name length
 Each folder is assigned a name, e.g., "Folder1". If the name length is not a multiple of 4 bytes, a minimum number of padding octets necessary to pad the ADU to a full 32 bit boundary follow the folder name.

The parent container (which is a folder) ID is encoded in the RTP/I state ADU.

4.1.1.2 Change Folder ADUs

For folders the following change operations exist (subtype given in brackets): set active (24), change name (23), and change parent object (27).

4.1.1.2.1 Change Name ADU



operation type (OPT) : 3 bits
 This field identifies the type of the ADU. It is set to 1 (change).

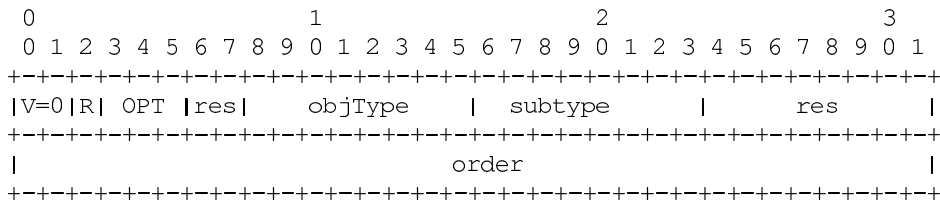
object type (objType) : 8 bits
 This field identifies the type of the object affected by the ADU.
 It is set to 1 (folder).

subtype : 8 bits
 This field is set to 23 (change name).

name length (nlength) : 8 bits
 This field identifies the length of the new folder name in bytes excluding any padding bytes. The maximum name length is 255.

folder name : length as specified in name length
 Each folder is assigned a name, e.g., "Folder1". If the name length is not a multiple of 4 bytes, a minimum number of padding octets necessary to pad the ADU to a full 32 bit boundary follow the folder name.

4.1.1.2.2 Change Parent Object ADU



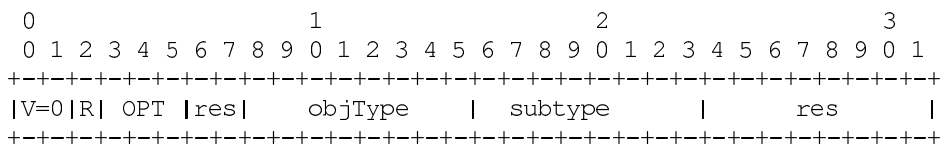
operation type (OPT) : 3 bits
 This field identifies the type of the ADU. It is set to 1 (change).

object type (objType) : 8 bits
 This field identifies the type of the object affected by the ADU.
 It is set to 1 (folder).

subtype : 8 bits
 This field is set to 27 (change parent object).

order : 32 bits
 This field denotes the ordering of objects at the same hierachy level.

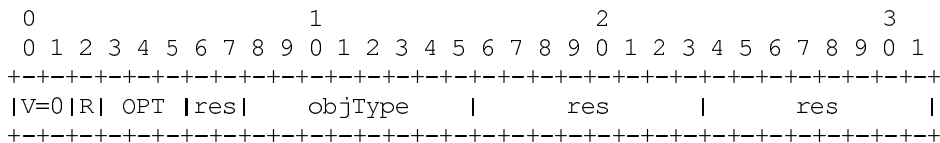
4.1.1.2.3 Set Active ADU



operation type (OPT) : 3 bits
 This field identifies the type of the ADU. It is set to 1 (change).

subtype : 8 bits
 This field is set to 24 (set active).

4.1.1.3 Delete Folder ADU



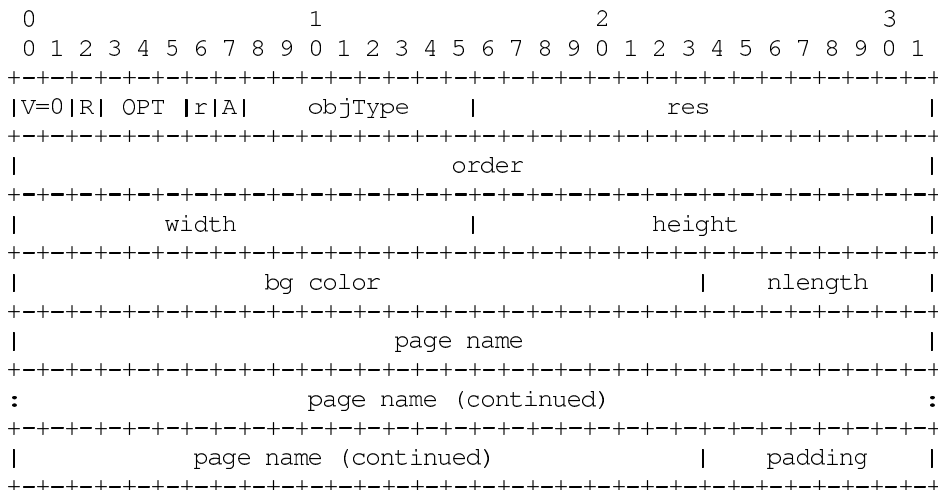
operation type (OPT) : 3 bits
 This field identifies the operation type of the ADU. It is set to 2 (delete).

object type (objType) : 8 bits
 This field identifies the type of the object affected by the ADU.
 It is set to 1 (folder).

4.1.2 Page ADUs

Pages have the object type (objType) 2.

4.1.2.1 Page State ADU



operation type (OPT) : 3 bits

This field identifies the type of the ADU. It is set to 0 (state).

reserved (r): 1 bit

Reserved for future use.

active (A) : 1 bit

This field identifies if the page is active (1) or not (0).

object type (objType) : 8 bits

This field identifies the type of the object affected by the ADU.
It is set to 2 (Page).

width : 16 bits

This field identifies the width of the page in pixels.

height : 16 bits

This field identifies the height of the page in pixels.

background color (bg color) : 24 bits

This field identifies the background color of the page.

name length (nlength) : 8 bits

This field identifies the length of the page name in bytes excluding any padding bytes. The maximum name length is 255.

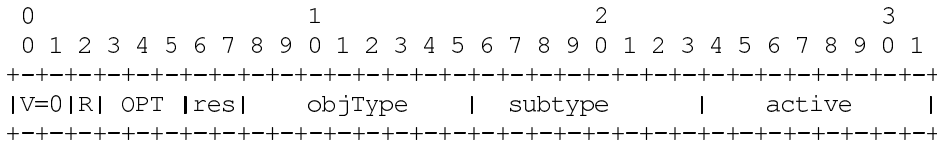
page name : length as specified in name length

Each page is assigned a name, e.g., "Page1". If the name length is not a multiple of 4 bytes, a minimum number of padding octets necessary to pad the ADU to a full 32 bit boundary follow the page name.

The parent container (which is a folder) ID is encoded in the RTP/I state ADU

4.1.2.2 Change Page ADUs For pages the following change operations exist (subtype given in brackets): set active (24), change size (2), change background color (22), change name (23), and change parent object (27).

4.1.2.2.1 Set Active ADU

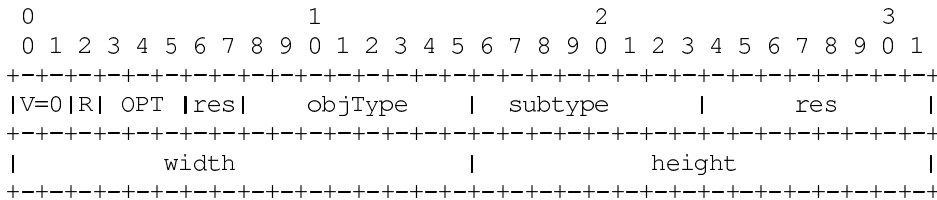


operation type (OPT) : 3 bits
This field identifies the type of the ADU. It is set to 1 (change).

subtype : 8 bits
This field is set to 24 (set active).

active : 8 bits
This field identifies if the page is active/visible (1) or not (0).

4.1.2.2.2 Change Size ADU



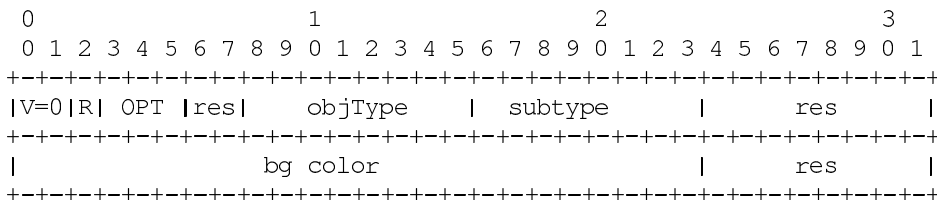
operation type (OPT) : 3 bits
This field identifies the type of the ADU. It is set to 1 (change).

subtype : 8 bits
This field is set to 1 (change size).

width : 16 bits
This field identifies the new width of the page in pixels.

height : 16 bits
This field identifies the new height of the page in pixels.

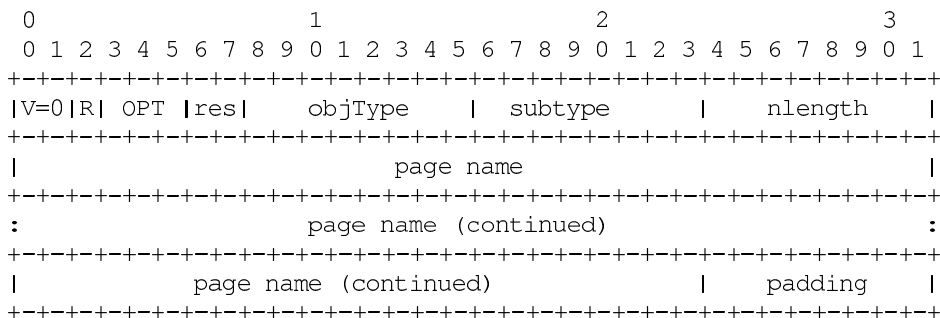
4.1.2.2.3 Change Background Color ADU



subtype : 8 bits
This field is set to 22 (change background color).

background color (bg color) : 24 bits
This field identifies the new background color of the page.

4.1.2.2.4 Change Name ADU

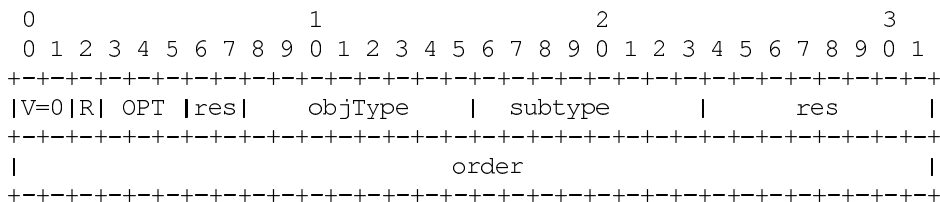


subtype : 8 bits
 This field is set to 23 (change name).

name length (nlength) : 8 bits
 This field identifies the new length of the page name in bytes excluding any padding bytes. The maximum name length is 255.

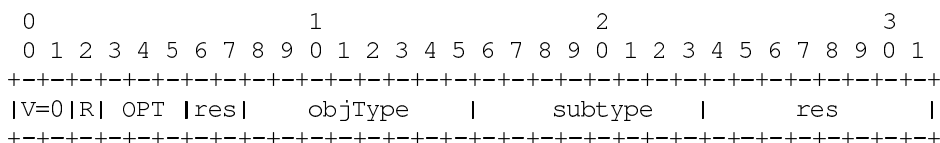
page name : length as specified in name length
 Each page is assigned a name, e.g., "Page1". If the name length is not a multiple of 4 bytes, a minimum number of padding octets necessary to pad the ADU to a full 32 bit boundary follow the page name.

4.1.2.2.5 Change Parent Object



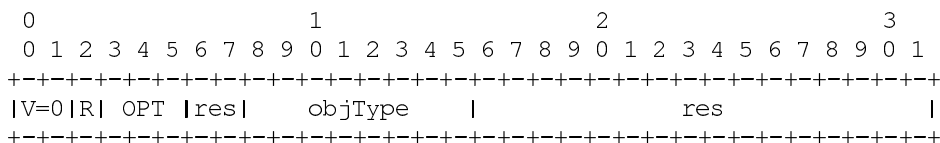
subtype : 8 bits
 This field is set to 27 (change parent object).

4.1.2.2.6 Set Active Item ADU



subtype : 8 bits
 This field is set to 28 (change active object). The new active item is given by the object ID encoded in the ADU Event Header.

4.1.2.3 Delete Page ADU

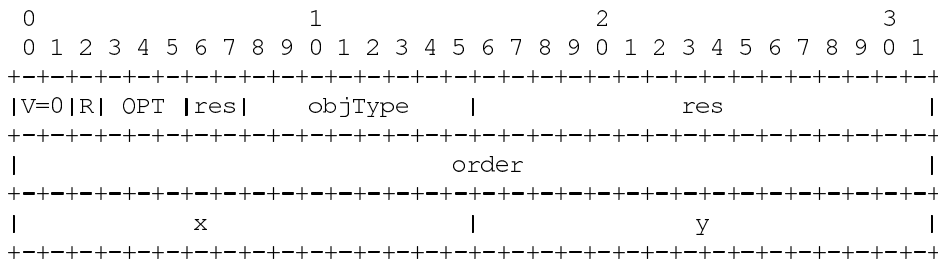


operation type (OPT) : 3 bits
 This field identifies the operation type of the ADU. It is set to 2 (delete).

4.1.3 Group ADUs

Groups have the object type (objType) 11.

4.1.3.1 Group State ADU



operation type (OPT) : 3 bits
This field identifies the type of the ADU. It is set to 0 (state).

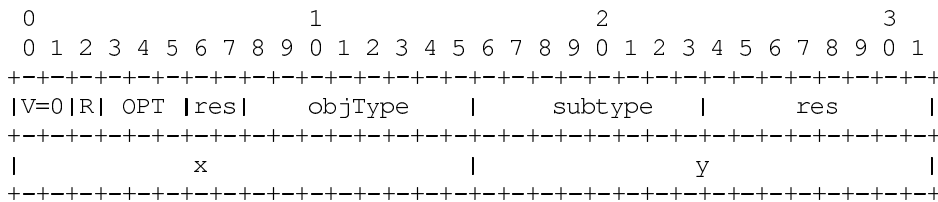
object type (objType) : 8 bits
This field identifies the type of the object affected by the ADU. It is set to 11 (group).

x-coordinate (x) : 16 bits
The x-coordinate identifies the horizontal position of the upper left corner of the object in pixels.

y-coordinate (y) : 16 bits
The y-coordinate identifies the vertical position of the upper left corner of the object in pixels.

4.1.3.2 Change Group ADUs For groups the following change operations exist (subtype given in brackets): move (1), and set active item (28).

4.1.3.2.1 Move ADU



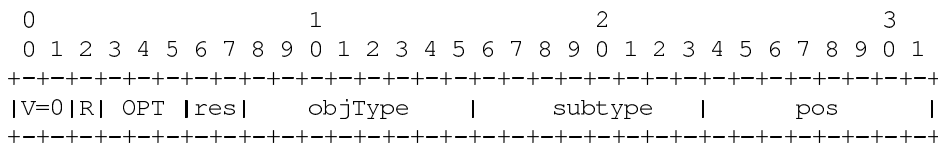
operation type (OPT) : 3 bits
This field identifies the type of the ADU. It is set to 1 (change).

subtype : 8 bits
This field is set to 1 (move object).

x-coordinate (x) : 16 bits
The x-coordinate identifies the new horizontal position of the upper left corner of the object in pixels.

y-coordinate (y) : 16 bits
The y-coordinate identifies the new vertical position of the upper left corner of the object in pixels.

4.1.3.2.2 Set Active Item ADU



subtype : 8 bits
 This field is set to 28 (change active object). The new active item is either given by the object ID encoded in the RTP/I event header or is given by the position field.

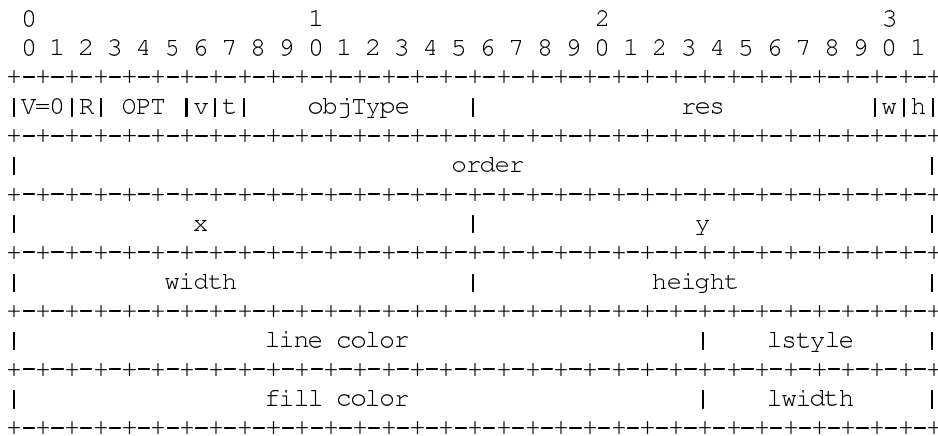
position (pos): 8 bits
 Set active item:
 0: the active item is identified by the object ID encoded in the RTP/I event header
 1: set first item active
 2: set last item active
 3: set active item one pos left in the object list of the group
 4: set active item one pos right in the object list of the group

4.2 Object ADUs

4.2.1 Rectangle ADUs

Rectangles have the object type (objType) 4.

4.2.1.1 Rectangle State ADU



operation type (OPT) : 3 bits
 This field identifies the type of the ADU. It is set to 0 (state).

visible (v) : 1 bit
 This bit is set, if the object represented by the ADU is visible. If the bit has the value 0, the object is created but not displayed.

transparent (t) : 1 bit
 This bit is set, if the object is outlined, i.e. the fillcolor of the object is transparent and shall not conceal any underlying objects.

object type (objType) : 8 bits
 This field identifies the type of the object affected by the ADU. It is set to 4 (rectangle).

sign of width (w) : 1 bit
 This field gives the sign of the width; if the bit is set (1), the width is negative, else positive, relative to the point given by the x- and y-coordinates.

sign of height (h) : 1 bit
 This field gives the sign of the height; if the bit is set (1), the height is negative, else positive, relative to the point given by the x- and y-coordinates.

x-coordinate (x) : 16 bits

The x-coordinate identifies the horizontal position of the first point set of the object in pixels.

y-coordinate (y) : 16 bits

The y-coordinate identifies the vertical position of the first point set of the object in pixels.

width : 16 bits

This field gives the width of the object in pixels.

height : 16 bits

This field gives the height of the object in pixels.

line color : 24 bits

This field identifies the line color which shall be used to display the line surrounding the object.

fill color : 24 bits

This field identifies the fill color which shall be used to fill the area enclosed by the object. If the transparent bit is set, this field has no meaning.

line style (lstyle) : 8 bits

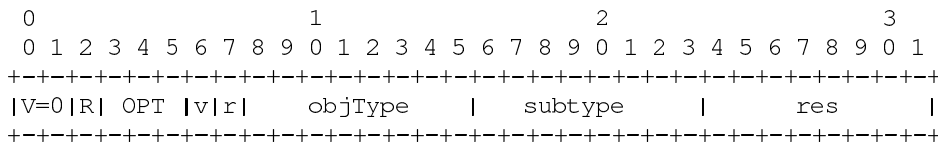
This field identifies the line style which shall be used to draw the line surrounding the object. Possible values are: solid (0) and dashed (1 - 6).

line width (lwidth) : 8 bits

This field identifies the line width which shall be used to draw the line surrounding the object.

4.2.1.2 Change Rectangle ADUs For rectangles the following change operations exist (subtype given in brackets): change visibility (0), move (1), change size (2), change line width (3), change line style (4), change line color (5), change fill color (6), raise (25), lower (26), and change parent (27).

4.2.1.2.1 Change Visibility ADU



operation type (OPT) : 3 bits

This field identifies the operation type of the ADU. It is set to 1 (change).

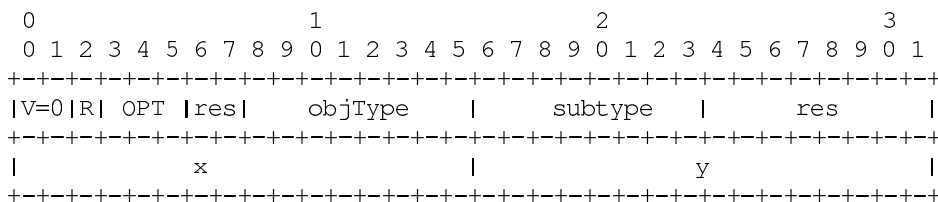
subtype : 8 bits

This field is set to 0 (change visibility).

visible (v) : 1 bit

This bit is set, if the object represented by the ADU is visible. If the bit has the value 0, the object becomes invisible. If the bit has the value 1, a former invisible object becomes visible.

4.2.1.2.2 Move ADU

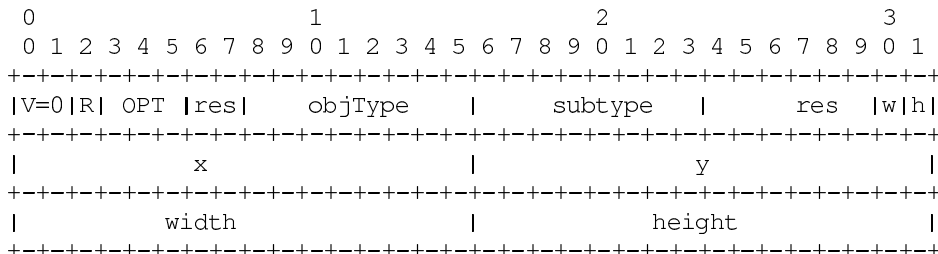


subtype : 8 bits
This field is set to 1 (move object).

x-coordinate (x) : 16 bits
The x-coordinate identifies the new horizontal position of the upper left corner of the object in pixels.

y-coordinate (y) : 16 bits
The y-coordinate identifies the new vertical position of the upper left corner of the object in pixels.

4.2.1.2.3 Change Size ADU



subtype : 8 bits
This field is set to 2 (change object size).

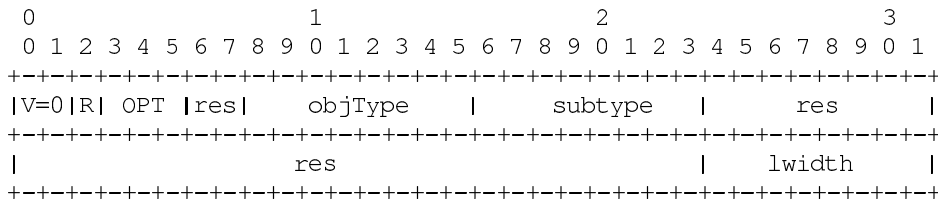
x-coordinate (x) : 16 bits
The x-coordinate identifies the new horizontal position of the upper left corner of the object in pixels.

y-coordinate (y) : 16 bits
The y-coordinate identifies the new vertical position of the upper left corner of the object in pixels.

width : 16 bits
This field gives the new width of the object in pixels.

height : 16 bits
This field gives the new height of the object in pixels.

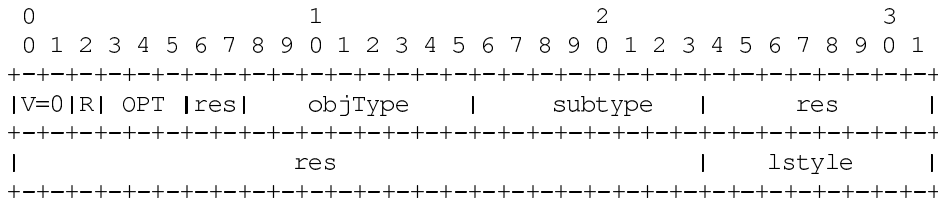
4.2.1.2.4 Change Line Width ADU



subtype : 8 bits
This field is set to 3 (change line width).

line width (lwidth) : 8 bits
This field identifies the new line width which shall be used to draw the line surrounding the object.

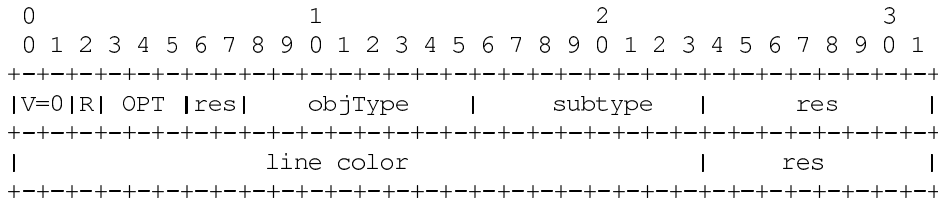
4.2.1.2.5 Change Line Style ADU



subtype : 8 bits
This field is set to 4 (change line style).

line style (lstyle) : 8 bits
This field identifies the new line style which shall be used to draw the line surrounding the object.

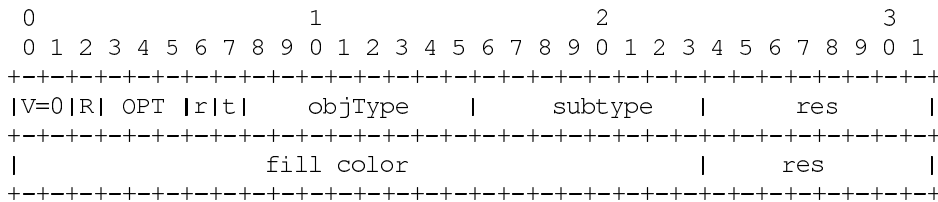
4.2.1.2.6 Change Line Color ADU



subtype : 8 bits
This field is set to 5 (change line color).

line color : 24 bits
This field identifies the new line color which shall be used to display the line surrounding the object.

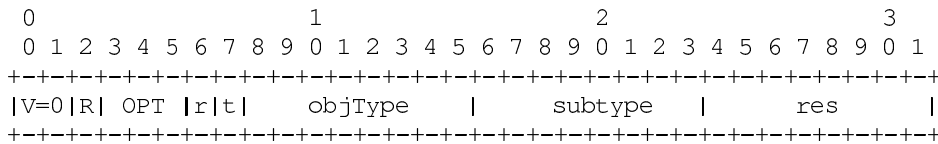
4.2.1.2.7 Change Fill Color ADU (Filled Rectangle only)



subtype : 8 bits
This field is set to 6 (change fill color).

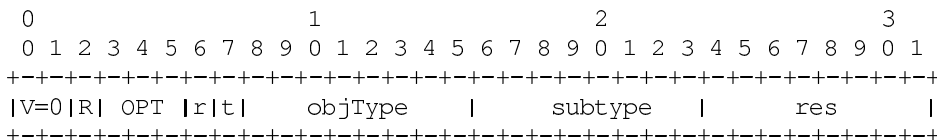
fill color : 24 bits
This field identifies the new fill color which shall be used to fill the area surrounded by the object.

4.2.1.2.8 Raise ADU



subtype : 8 bits
This field is set to 25 (raise). A raise operation changes the display order of the parent page so that the object is not (partly) covered by all other objects in the same area.

4.2.1.2.9 Lower ADU

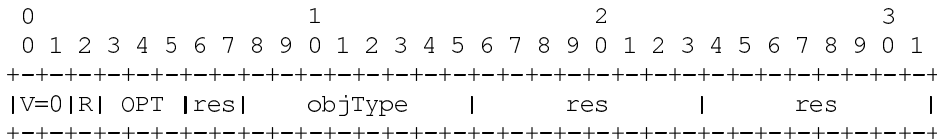


subtype : 8 bits

This field is set to 26 (lower). A lower operation changes the display order of the parent page so that the object is (partly) covered by all other objects in the same area.

4.2.1.2.10 Change Parent Object ADU

4.2.1.3 Delete Rectangle ADU



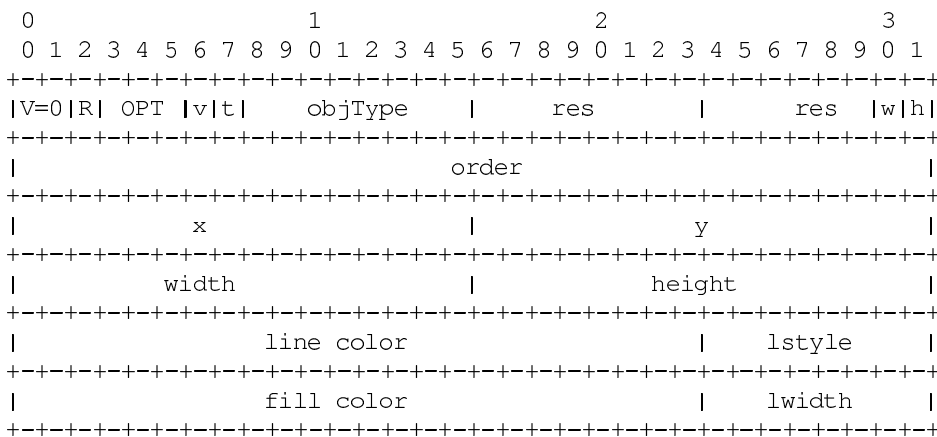
operation type (OPT) : 3 bits

This field identifies the operation type of the ADU. It is set to 2 (delete).

4.2.2 Oval ADUs

Ovals have the object type (objType) 5.

4.2.2.1 Oval State ADU



operation type (OPT) : 3 bits

This field identifies the operation type of the ADU. It is set to 0 (state).

object type (objType) : 8 bits

This field identifies the type of the object affected by the ADU. It is set to 5 (oval).

4.2.2.2 Change Oval ADUs For ovals the following change operations exist (subtype given in brackets): change visibility (0), move (1), change size (2), change line width (3), change line style (4), change line color (5), change fill color (6), raise(25), lower (26), and change parent (27).

4.2.2.2.1 Change Visibility ADU

4.2.2.2.2 Move ADU

4.2.2.2.3 Change Size ADU

4.2.2.2.4 Change Line Width ADU

4.2.2.2.5 Change Line Style ADU

4.2.2.2.6 Change Line Color ADU

4.2.2.2.7 Change Fill Color ADU

4.2.2.2.8 Raise ADU

4.2.2.2.9 Lower ADU

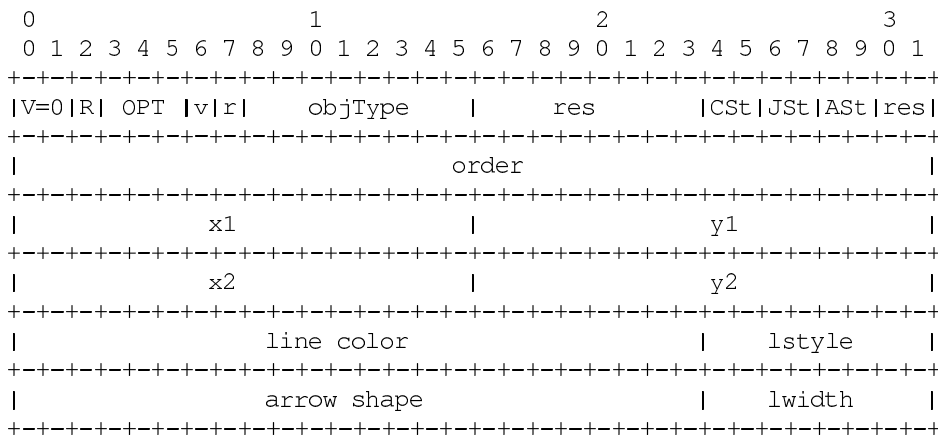
4.2.2.2.10 Change Parent Object ADU

4.2.2.3 Delete Oval ADU

4.2.3 Line ADUs

Lines have the object type (objType) 6.

4.2.3.1 Line State ADU



operation type (OPT) : 3 bits
This field identifies the operation type of the ADU. It is set to 0 (state).

object type (objType) : 8 bits
This field identifies the type of the object affected by the ADU. It is set to 6 (line).

cap-style (CSt) : 2 bits
This field identifies the cap-style of the line to be drawn. Possible values are 0 (butt), 1 (round) and 2 (projecting). Value 3 is not possible.

join-style (JSt) : 2 bits
This field identifies the join-style of the line to be drawn. Possible values are 0 (bevel), 1 (miter) and 2 (round). Value 3 is not possible.

arrow-style (ASt) : 2 bits
This field identifies the arrow-style of the line to be drawn. Possible values are 0 (none), 1 (first), 2 (last) and 3 (both).

point 1 x-coordinate (x1) : 16 bits
The x1-coordinate identifies the horizontal position of the first point of the line.

point 1 y-coordinate (y1) : 16 bits

The y1-coordinate identifies the vertical position of the first point of the line.

point 2 x-coordinate (x2) : 16 bits

The x2-coordinate identifies the horizontal position of the second point of the line.

point 2 y-coordinate (y2) : 16 bits

The y2-coordinate identifies the vertical position of the second point of the line.

arrow shape : 24 bits

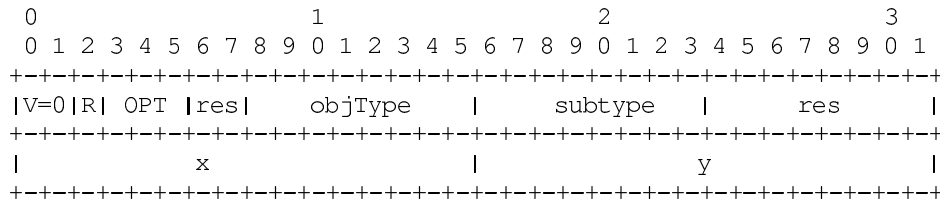
This field identifies the arrow shape (the line ending) of the line to be drawn. It is a triple of 8 bit values shaping an arrow in pixels. The arrow shape is given only if the arrow-style is unequal 0.

4.2.3.2 Change Line ADUs For lines the following change operations exist (subtype given in brackets): change visibility (0), move (1), move point (7/8), change line width (3), change line style (4), change line color (5), change cap-style (9), change join-style (10), change arrow-style (11), raise (25), lower (26), and change parent (27).

4.2.3.2.1 Change Visibility ADU

4.2.3.2.2 Move ADU

4.2.3.2.3 Move Point ADU



operation type (OPT) : 3 bits

This field identifies the operation type of the ADU. It is set to 1 (change).

object type (objType) : 8 bits

This field identifies the type of the object affected by the ADU. It is set to 6 (line).

subtype : 8 bits

This field is set to 7 (change point 1 position) or 8 (change point 2 position).

x-coordinate (x) : 16 bits

This field gives the new horizontal position of the affected point (1 or 2) in pixels.

y-coordinate (y) : 16 bits

This field gives the new vertical position of the affected point (1 or 2) in pixels.

4.2.3.2.4 Change Line Width ADU

4.2.3.2.5 Change Line Style ADU

4.2.3.2.6 Change Line Color ADU

4.2.3.2.7 Change Cap-style ADU

```
0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+
|V=0|R| OPT |res|  objType  |  subtype  |CSt|  res  |
+-----+-----+-----+-----+
```

subtype : 8 bits

This field is set to 9 (change cap-style)

cap-style (CSt) : 2 bits

This field identifies the new cap-style of the line to be drawn. Possible values are 0 (butt), 1 (round) and 2 (projecting). Value 3 is not possible.

4.2.3.2.8 Change Join-style ADU

```
0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+
|V=0|R| OPT |res|  objType  |  subtype  |res|JSt|  res  |
+-----+-----+-----+-----+
```

subtype : 8 bits

This field is set to 10 (change join-style)

join-style (JSt) : 2 bits

This field identifies the new join-style of the line to be drawn. Possible values are 0 (bevel), 1 (miter) and 2 (round). Value 3 is not possible.

4.2.3.2.9 Change Arrow-style ADU

```
0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+
|V=0|R| OPT |res|  objType  |  subtype  | res |ASt|res|
+-----+-----+-----+-----+
|                arrow shape                |  res  |
+-----+-----+-----+-----+
```

subtype : 8 bits

This field is set to 11 (change arrow-style)

arrow-style (ASt) : 2 bits

This field identifies the new arrow-style of the line to be drawn. Possible values are 0 (none), 1 (first), 2 (last) and 3 (both).

arrow shape : 24 bits

This field identifies the new arrow shape (the line ending) of the line to be drawn. It is a triple of 8 bit values shaping an arrow in pixels.

4.2.3.2.10 Raise ADU

4.2.3.2.11 Lower ADU

4.2.3.2.12 Change Parent Object ADU

4.2.3.3 Delete Line ADU

4.2.4 Polygon ADUs

Polygons have the object type (objType) 8.

4.2.4.1 Polygon State ADU At first a polygon exists of exactly one point given by (x,y). Additional points are added by using add point ADUs. The drawing process is finished by a close polygon ADU.

0			1							2							3																											
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9					
V=0 R			OPT							v t							objType														res							w h						
			order																																									
			width														height																											
			line color														lstyle																											
			fill color														lwidth																											
			count														res																											
			x (1)														y (1)																											
			x (2)														y (2)																											
:			...							:							:							:							:													
			x (count)														y (count)																											

operation type (OPT) : 3 bits
This field identifies the type of the ADU. It is set to 0 (state).

object type (objType) : 8 bits
This field identifies the type of the object affected by the ADU.
It is set to 8 (polygon).

count : 16 bits
This field identifies the number of points that define the polygon:
the points are connected by lines of the style encoded in the ADU.
Following, an ordered list of all points is given.

4.2.4.2 Change Polygon ADUs For polygons the following change operations exist (subtype given in brackets): change visibility (0), move (1), move point (14), change line width (3), change line style (4), change line color (5), change fill color (6), add point (12), close polygon (29), raise (25), lower (26), and change parent (27).

4.2.4.2.1 Change visibility ADU

4.2.4.2.2 Move ADU

4.2.4.2.3 Move Point ADU

0			1							2							3																																									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9																			
V=0 R			OPT							res							objType														subtype														res													
			pointID														res																																									
			x														y																																									

operation type (OPT) : 3 bits
This field identifies the operation type of the ADU. It is set to 1 (change).

subtype : 8 bits
This field is set to 14 (move point).

point identifier (pointID) : 16 bits
This field identifies the point which is affected by the move operation. The first point has the ID 0.

x-coordinate (x) : 16 bits
This field gives the new horizontal position of the affected point in pixels.

y-coordinate (y) : 16 bits
This field gives the new vertical position of the affected point in pixels.

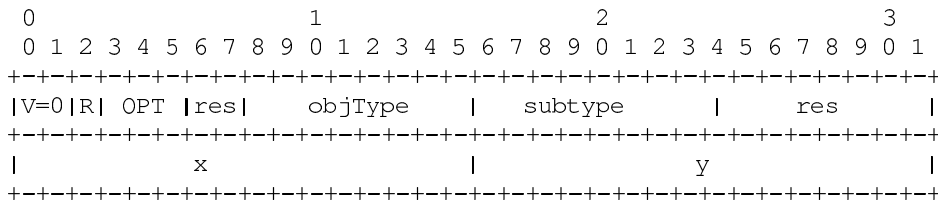
4.2.4.2.4 Change Line Width ADU

4.2.4.2.5 Change Line Style ADU

4.2.4.2.6 Change Line Color ADU

4.2.4.2.7 Change Fill Color ADU

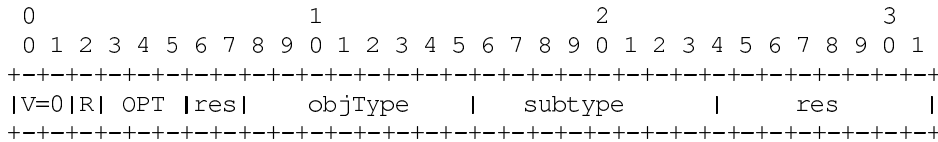
4.2.4.2.8 Add Point ADU



operation type (OPT) : 3 bits
This field identifies the operation type of the ADU. It is set to 1 (change).

subtype : 8 bits
This field is set to 12 (add Point).

4.2.4.2.9 Close Polygon ADU



subtype : 8 bits
This field is set to 29 (close polygon). This event denotes that the drawing process has been finished (i.e., there are no more points to be added) and the line between first and last point is to be drawn.

4.2.4.2.10 Raise ADU

4.2.4.2.11 Lower ADU

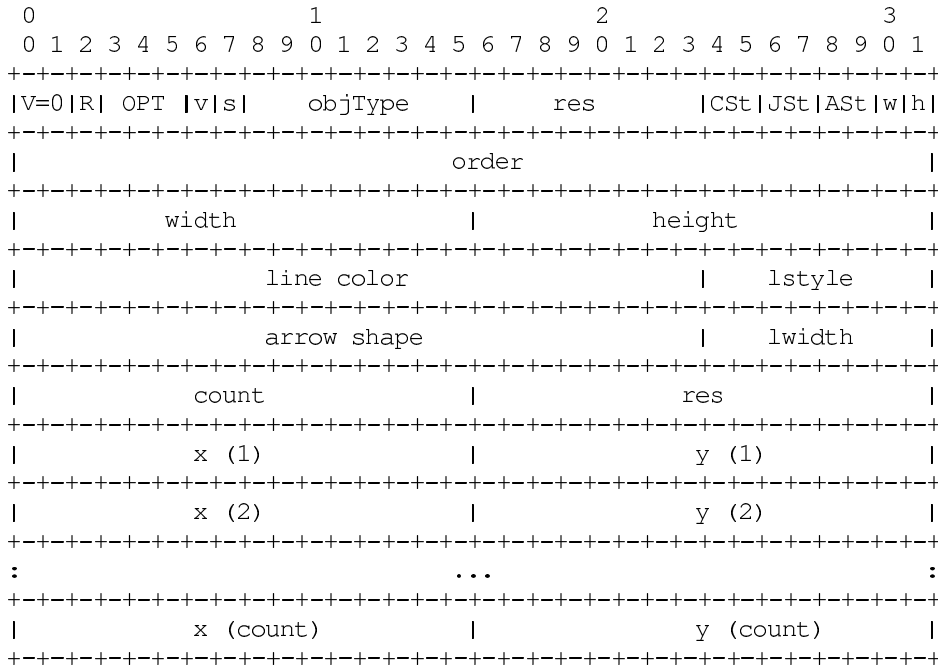
4.2.4.2.12 Change Parent Object ADU

4.2.4.3 Delete Polygon ADU

4.2.5 Polyline ADUs

Polylines have the object type (objType) 7.

4.2.5.1 Polyline State ADU



smooth (s) : 1 bit

If this bit is set, the polyline shall be displayed smoothed.

operation type (OPT) : 3 bits

This field identifies the operation type of the ADU. It is set to 0 (state).

object type (objType) : 8 bits

This field identifies the type of the object affected by the ADU.

It is set to 7 (polyline).

4.2.5.2 Change Polyline ADUs For polylines the following change operations exist (subtype given in brackets): change visibility (0), move (1), move point (14), change line width (3), change line style (4), change line color (5), change cap style (9), change join style (10), change arrow style (11), add Point (12), raise (25), lower (26), and change parent (27).

4.2.5.2.1 Change Visibility ADU

4.2.5.2.2 Move ADU

4.2.5.2.3 Move Point ADU

4.2.5.2.4 Change Line Width ADU

4.2.5.2.5 Change Line Style ADU

4.2.5.2.6 Change Line Color ADU

4.2.5.2.7 Change Cap-style ADU

4.2.5.2.8 Change Join-style ADU

4.2.5.2.9 Change Arrow-style ADU

4.2.5.2.10 Add Point ADU

4.2.5.2.11 Raise ADU

4.2.5.2.12 Lower ADU

4.2.5.2.13 Change Parent Object ADU

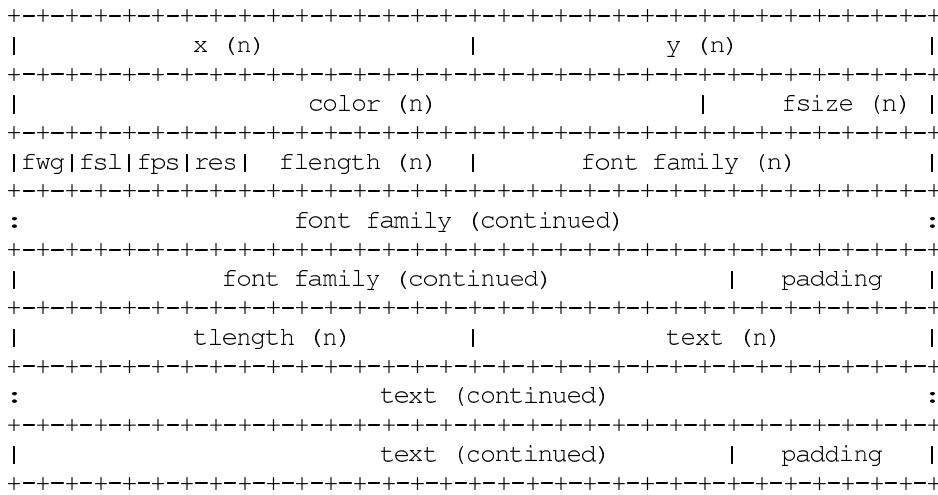
4.2.5.3 Delete Polyline ADU

4.2.6 Text ADUs

Text ADUs have the object type (objType) 9.

4.2.6.1 Text State ADU

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
V=0 R OPT v r										objType										res										nbr									
										order																													
										width										height																			
active text										cpos										cursor index																			
x (1)																				y (1)																			
										color (1)										fsize (1)																			
fwg fsl fps res										flength (1)										font family (1)																			
:										font family (continued)										:																			
										font family (continued)										padding																			
										tlength (1)										text (1)																			
:										text (continued)										:																			
										text (continued)										padding																			
										x (2)										y (2)																			
										color (2)										fsize (2)																			
fwg fsl fps res										flength (2)										font family (2)																			
:										font family (continued)										:																			
										font family (continued)										padding																			
										tlength (2)										text (2)																			
:										text (continued)										:																			
										text (continued)										padding																			
:																				:																			



operation type (OPT) : 3 bits
 This field identifies the type of the ADU. It is set to 0 (state).

object type (objType) : 8 bits
 This field identifies the type of the object affected by the ADU.
 It is set to 9 (text).

number of text elements (nbr) : 8 bits
 A Text can consist of different text elements with different font attributes and positions. The nbr field gives the number of encoded elements.

active text : 8 bits
 This field identifies the index of the active text element starting by 0.

cursor position (cpos): 8 bits
 This field identifies if the shared cursor is to be displayed at a special position at the active text element:
 0: the cursor position is identified by the cursor index
 1: the cursor is to be displayed before the first character
 2: no shared cursor

cursor index : 16 bits
 This field contains the index of the active character starting by 0; the text cursor is to be displayed one position behind this character.

color : 24 bits
 This field identifies the font color of the Text.

font size (fsize) : 8 bits
 This field identifies the font size of the text.

font weight (fwg) : 2 bits
 This field identifies the font weight of the text. Possible values are normal (0) and bold (1).

font slant (fsl) : 2 bits
 This field identifies the font slant of the text. Possible values are roman (0) and italic (1).

font position (fps) : 2 bits
 This field identifies the font position of the text. Possible values are normal (0), high (1) and low (2).

font family length (flength) : 8 bits
 This field identifies the length of the font family in bytes excluding any padding octets.

font family : length as specified in font family length
 The font family is encoded as string. Examples for the font family are "Times New Roman", and "Arial". If the name of the font family is not a multiple of 4 bytes, a minimum number of padding octets necessary to pad the ADU to a full 32 bit boundary follow the font family name.

text length (tlength) : 16 bits
 This field identifies the length of the text in bytes excluding any padding octets.

text : length as specified in text length
 The text is encoded as ascii string. If the text is not a multiple of 4 bytes, a minimum number of padding octets necessary to pad the text item to a full 32 bit boundary follow the text.

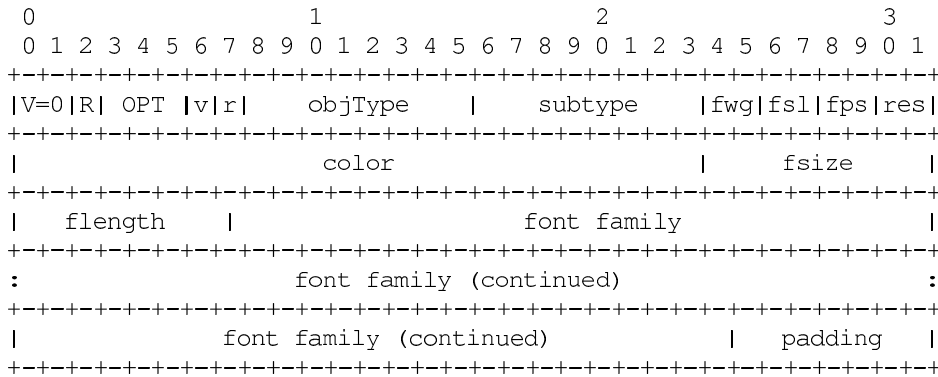
4.2.6.2 Change Text ADUs For text the following change operations exist (subtype given in brackets): change visibility (0), move (1), change size (2), change font (30), insert new font (31), insert new font color (32), change font family (15), change font size (16), change font color (5), change font weight (17), change font slant (18), change font position (19), insert characters (20), delete characters (21), raise (25), lower (26), and change parent (27).

4.2.6.2.1 Change Visibility ADU

4.2.6.2.2 Move ADU

4.2.6.2.3 Change Size ADU

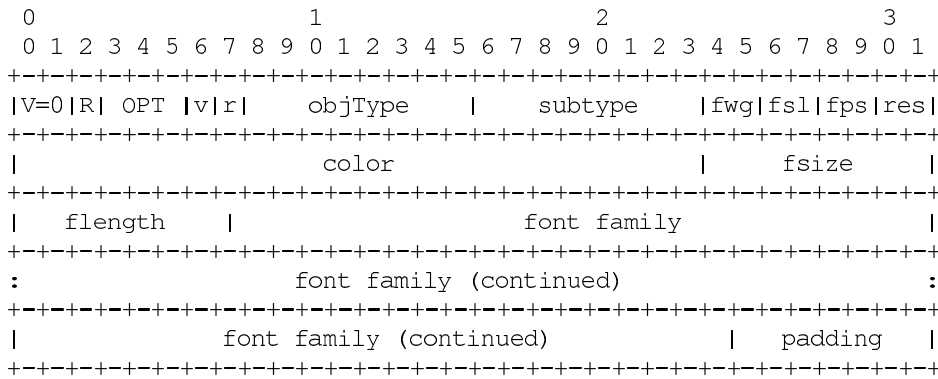
4.2.6.2.4 Change Font ADU



operation type (OPT) : 3 bits
 This field identifies the type of the ADU. It is set to 1 (change).

subtype : 8 bits
 This field is set to 30 (change font). The operation changes all elements of the text to the given font.

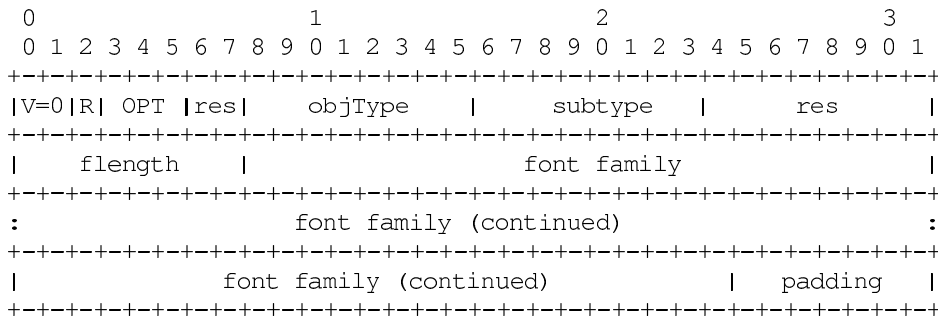
4.2.6.2.5 Insert Font ADU



operation type (OPT) : 3 bits
This field identifies the type of the ADU. It is set to 1 (change).

subtype : 8 bits
This field is set to 31 (insert font). The operation inserts a new text element at the active cursor position with the given font.

4.2.6.2.6 Change Font Family ADU



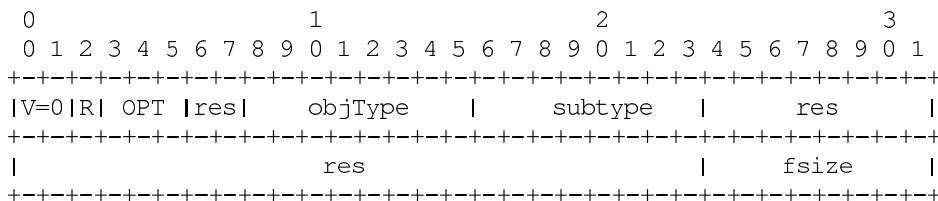
operation type (OPT) : 3 bits
This field identifies the type of the ADU. It is set to 1 (change).

subtype : 8 bits
This field is set to 15 (change font family). The operation changes all elements of the text to the given font family.

font family length (flength) : 8 bits
This field identifies the length of the new font family in bytes excluding any padding octets.

font family : length as specified in font family length
The font family is encoded as string. If the name of the font family is not a multiple of 4 bytes, a minimum number of padding octets necessary to pad the ADU to a full 32 bit boundary follow the font family name.

4.2.6.2.7 Change Font Size ADU



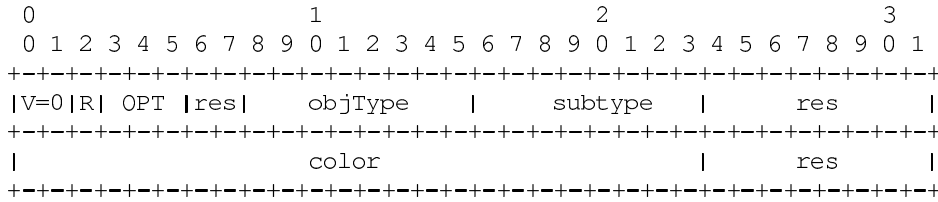
subtype : 8 bits
This field is set to 16 (change font size). The operation changes all

elements of the text to the given font size.

font size (fsize) : 8 bits

This field identifies the new font size of the text.

4.2.6.2.8 Change Font Color ADU



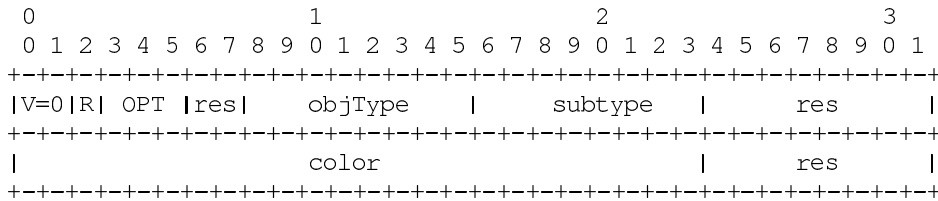
subtype : 8 bits

This field is set to 5 (change font color). This operation changes the color of all text elements.

color : 24 bits

This field identifies the new font color of the text.

4.2.6.2.9 Insert New Font Color ADU



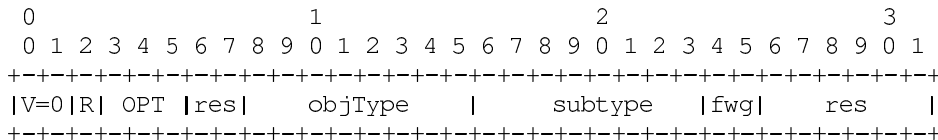
subtype : 8 bits

This field is set to 32 (insert new font color). This operation inserts a new text element with the given color.

color : 24 bits

This field identifies the font color of the text.

4.2.6.2.10 Change Font Weight ADU



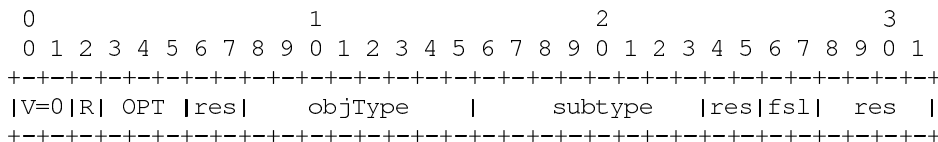
subtype : 8 bits

This field is set to 17 (change font weight). The operation changes all elements of the text to the given font weight.

font weight (fwg) : 2 bits

This field identifies the new font weight of the text. Possible values are normal (0) and bold (1).

4.2.6.2.11 Change Font Slant ADU



subtype : 8 bits

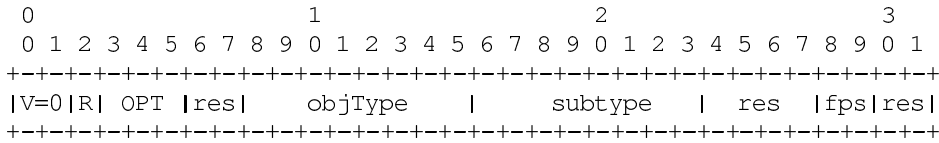
This field is set to 18 (change font slant). The operation changes all

elements of the text to the given font slant.

font slant (fsl) : 2 bits

This field identifies the new font slant of the text. Possible values are roman (0) and italic (1).

4.2.6.2.12 Change Font Position ADU



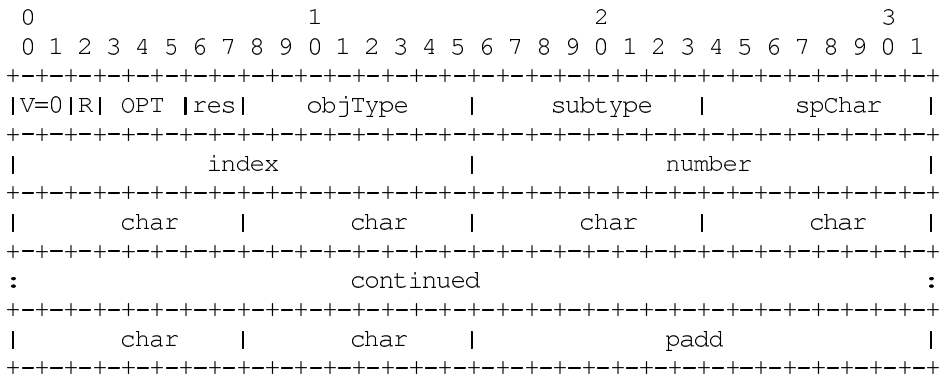
subtype : 8 bits

This field is set to 19 (change font position). The operation changes all elements of the text to the given font position.

font position (fps) : 2 bits

This field identifies the new font position of the text. Possible values are normal (0), high (1) and low (2).

4.2.6.2.13 Insert Characters ADU



subtype : 8 bits

This field is set to 20 (insert character).

special character (spChar) : 8 bits

If not equal 0, a special character is to be inserted at the specified position index. Then number must be 0 and the char list empty (ADU length 8 octets). Possible values for the special character are return (1).

index : 16 bits

This field identifies the starting index where the charactes shall be inserted into the text string. The first character has the index 0.

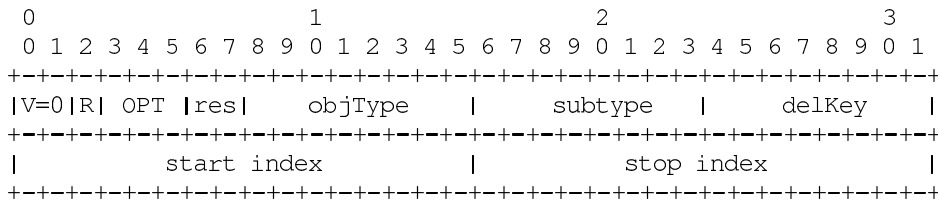
number : 16 bits

This field identifies the number of characters to be inserted into the text string, excluding any padding octets that might be necessary to padd the ADU to a full 32 bit border.

character (char) : 8 bits

Each character is encoded as ascii. The first character is to be inserted into the text string at position index.

4.2.6.2.14 Delete Characters ADU



subtype : 8 bits

This field is set to 21 (delete character).

delKey : 8 bits

If not equal 0, this field encodes the function of a deletion key:

1: <Backspace>, delete char at current cursor position

2: <Delete>, delete char following the current cursor position

start index : 16 bits

This field identifies the index of the first character that shall be deleted in the text string. The first character in the text string has the index 0.

stop index : 16 bits

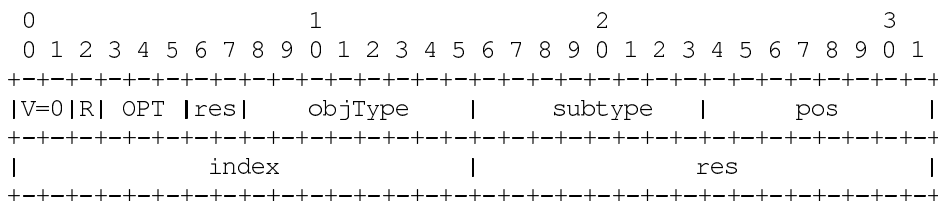
This field identifies the index of the last character that shall be deleted in the text string (including).

4.2.6.2.15 Raise ADU

4.2.6.2.16 Lower ADU

4.2.6.2.17 Change Parent Object ADU

4.2.6.2.18 Set Active Item ADU (Position Text Cursor ADU) The ADU indicates where to set a shared text cursor. This means that that all participants share the same cursor and joint editing is possible only at the position indicated by the cursor.



subtype : 8 bits

This field is set to 28 (change active object).

Position (pos) : 8 bits

Set position of text cursor given by pos whereby the cursor is drawn behind the active char:

0: the cursor position is given by index

1: set cursor to position before first character (equaling <Home>-Key)

2: set cursor to position behind last char (equaling <End>-Key)

3: set cursor one position to the left

4: set cursor one position to the right

index (optional) : 16 bits

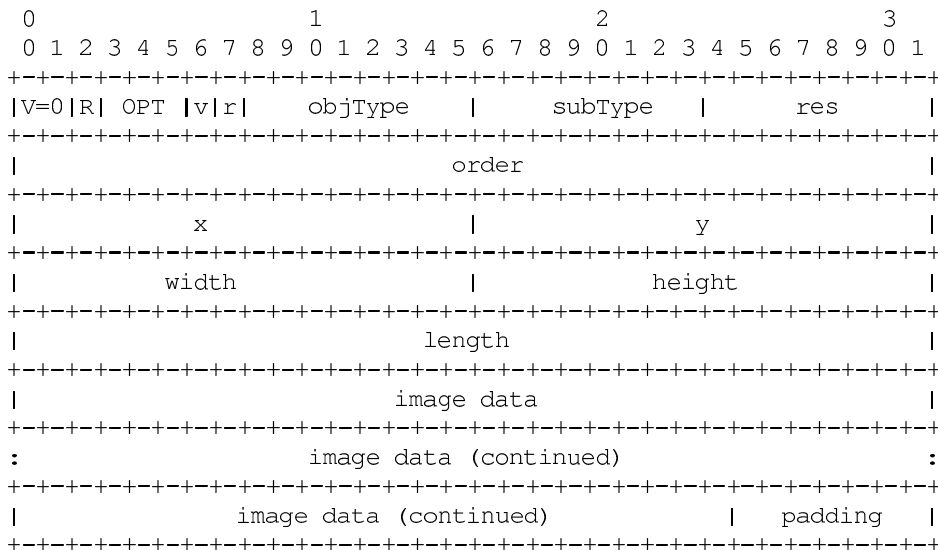
This field gives the index of the active character and identifies the position of the text cursor (one behind active).

4.2.6.3 Delete Text ADU

4.2.7 Image ADUs

Image ADUs have the object type (objType) 10.

4.2.7.1 Image State ADU



operation type (OPT) : 3 bits
This field identifies the operation type of the ADU. It is set to 0 (state).

object type (objType) : 8 bits
This field identifies the type of the object affected by the ADU.
It is set to 11 (Image).

subtype : 8 bits
This field identifies the image coding type. Possible values are:
GIF (0), JPEG (2) and postscript(3). Other types are possible, too.

length : 32 bits
This field identifies the length of the image data in bytes excluding any padding octets.

image data : length as specified in length
The image data encoded as specified by subtype. If the data length is not a multiple of 4 bytes, a minimum number of padding octets necessary to pad the ADU to a full 32 bit boundary follow the image data.

4.2.7.2 Change Image ADUs For images the following change operations exist (subtype given in brackets): change visibility (0), move (1), change size (2), raise (25), lower (26), and change parent (27).

4.2.7.2.1 Change Visibility ADU

4.2.7.2.2 Move ADU

4.2.7.2.3 Change Size ADU

4.2.7.2.4 Raise ADU

4.2.7.2.5 Lower ADU

4.2.7.2.6 Change Parent Object ADU

4.2.7.3 Delete Image ADU

5 Reserved Numbers

5.1 ADU Types

- 0 state
- 1 change
- 2 delete

5.2 Object Types

- 0 session
- 1 folder
- 2 page
- 3 layer
- 4 rectangle
- 5 oval
- 6 line
- 7 polyline
- 8 polygon
- 9 text
- 10 image
- 11 group

5.3 Change Types

- 0 change visibility
- 1 move
- 2 change size
- 3 change line width
- 4 change line style
- 5 change line color
- 6 change fill color
- 7 move point (1)
- 8 move point (2)
- 9 change cap-style
- 10 change join-style
- 11 change arrow-style
- 12 add point
- 29 close polygon
- 13 change type
- 14 move point
- 30 change font
- 31 insert new font
- 32 insert new font color
- 15 change font family
- 16 change font size
- 17 change font weight
- 18 change font slant
- 19 change font position
- 20 insert character
- 21 delete character
- 22 change background color
- 23 change name
- 24 set active
- 25 raise
- 26 lower
- 27 change parent object
- 28 change active object

References

- [GD98] L. Gautier and C. Diot. Design and Evaluation of MiMaze, a Multi-player Game on the Internet. In *IEEE International Conference on Multimedia Computing and Systems*, pages 233–236, 1998.
- [Gey99] W. Geyer. *Das digital lecture board – Konzeption, Design und Entwicklung eines Whiteboards für synchrones Teleteaching*. PhD thesis, University of Mannheim, Germany, 1999.
- [Hag96] O. Hagesand. Interactive multiuser VEs in the DIVE system. *IEEE Multimedia*, 3(1):30–39, 1996.
- [HC97] M. Handley and J. Crowcroft. Network Text Editor (NTE) – A scalable shared text editor for the MBone. In *ACM SIGCOMM*, pages 197–208, 1997.
- [Mau00] M. Mauve. *Distributed Interactive Media*. PhD thesis, University of Mannheim, Germany, 2000.
- [MHK⁺00] M. Mauve, V. Hilt, C. Kuhmünch, J. Vogel, W. Geyer, and W. Effelsberg. RTP/I: An Application-Level Real-Time Protocol for Distributed Interactive Media. Internet Draft: draft-mauve-rtpi-00.txt, 2000.
- [MHKE01] M. Mauve, V. Hilt, C. Kuhmüch, and W. Effelsberg. RTP/I - Toward a Common Application-Level Protocol for Distributed Interactive Media. *IEEE Transactions on Multimedia*, 3(1), 2001.
- [Tun98] T.L. Tung. MediaBoard. Master’s thesis, University of California, Berkely, California, USA, 1998.