REIHE INFORMATIK
9/2001

RTP/I Payload Type Definition for Telepointers

Jürgen Vogel
Universität Mannheim
Praktische Informatik IV
L15, 16
D-68131 Mannheim

RTP/I Payload Type Definition for Telepointers

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 telepointers 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 telepointer creates a common point of reference in distributed (i.e., multi-user) applications by visualizing mouse movements of remote session participants. Telepointers are used in conjunction with other distributed interactive media such as shared whiteboards and distributed virtual environments. This document specifies how to employ two-dimensional telepointers with RTP/I and defines application data units (ADUs) for telepointer operations. This protocol definition allows standardized collaboration between different telepointer 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]. Many of these make use of telepointers to support collaboration between session participants by visualizing mouse pointer movements of remote users. Despite being deployed in conjunction with other applications, telepointers are an independent type of distributed interactive media. 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 two-dimensional telepointer includes a tuple of screen coordinates.

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 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*.

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 the semantics of the transmitted data to a large extent 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. it describes how the medium-specific data are encoded and specifies a payload for telepointers. The aim of such a standardized protocol is to allow communication between different telepointer implementations. The specific encoding ”telepointer” as defined by this document is assigned the payload type ”4”. 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 for telepointers. Then we define all the necessary application data units (ADUs) of a telepointer protocol. These ADUs are transported either as an 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 telepointer consists of a type identifying the style of the telepointer (e.g., ”arrow pointing to the upper left corner of the screen”), of a tuple of coordinates (x,y) in pixels denoting the screen position the telepointer points to, and a color in which the telepointer is to be displayed. Please note that this document covers only two-dimensional telepointers that are used, e.g., with shared whiteboards. For applications such as distributed virtual environments a three-dimensional telepointer would be necessary implying a different payload type definition.

At a certain point of time, each user can control at most one telepointer. However, it is possible to have more than one active telepointer controlled by different session participants simultaneously. Every telepointer 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]. For better differentiation, telepointers should be displayed in different colors or types.

Each user interaction that changes the state of the telepointer (e.g., moving its position or changing its color) is translated to an ADU that is distributed via RTP/I as state, event or cue. Creating a telepointer causes transmission of an state. Changing attributes (e.g., color, position) of a telepointer or deleting a telepointer 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.

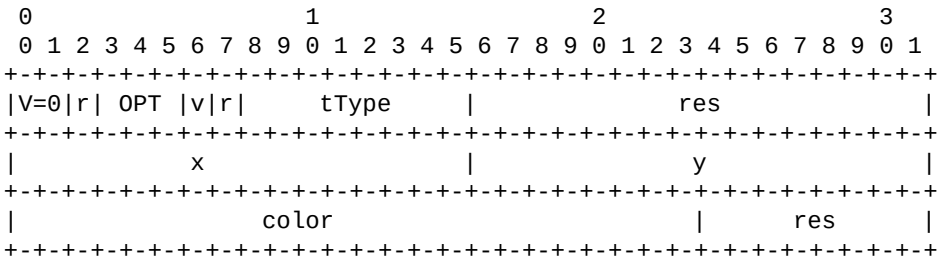
At a certain point in time, a telepointer is controlled by exactly one session participant. A state query therefore needs to be answered by this controlling participant; other participants should ignore the query.

All telepointers visible must be marked as active sub-components[MHK+00].

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 (reliable transport of cues is not necessary). This mechanism can be integrated either into the application, or into RTP/I, or can be implemented at the transport level.

3 Telepointer ADUs

3.1 Telepointer State ADU



version (V) : 2 bits
: This field specifies the version number of this protocol. The version defined by this specification is version 0.

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.

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.

telepointer type (tType) : 8 bits
: This field identifies the appearance type of the telepointer. The appearance type is defined as compass orientation. The orientation denotes the pointing direction of the telepointer. 0 is for east, 1 is for north, 2 is for west, 3 is for south, 4 is for north-east, 5 is for south-east, 6 is for south-west, 7 is for north-west. Other types are possible.

x-coordinate (x) : 16 bits
: The x-coordinate identfies the horizontal position of the pixel that is highlighted by the telepointer.

y-coordinate (y) : 16 bits
: The y-coordinate identfies the vertical position of the pixel that is highlighted by the telepointer.

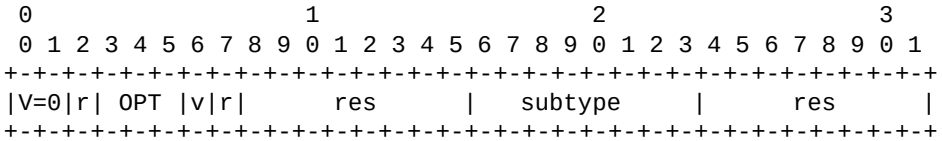
color : 24 bits
: This field identifies the color of the Telepointer. The color is encoded as 24-bit RGB.

reserved (r and res) : x bits
: This fields are reserved for future usage.

3.2 Change Telepointer ADU

For telepointers the following change operations exist (subtype given in brackets): change visibility (0), move (1), change type (2) and change color (3).

3.2.1 Change visibility ADU

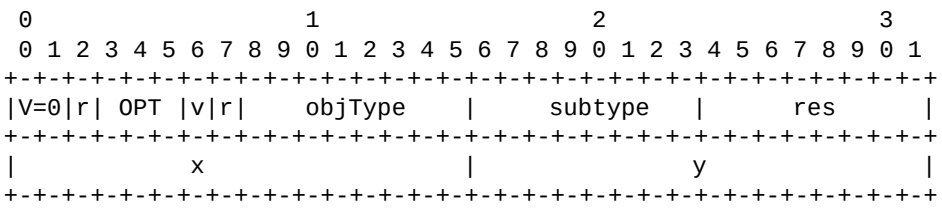


operation type (OPT) : 3 bits
     This field identifies the operation type of the ADU. It is set to 1 (change).

visible (v) : 1 bit
     This bit is set, if the telepointer represented by the ADU is visible. If the bit has the value 0, the telepointer becomes invisible. If the bit has the value 1, a former invisible telepointer becomes visible.

subtype : 8 bits
     This field is set to 0 (change visibility).

3.2.2 Move Telepointer ADU



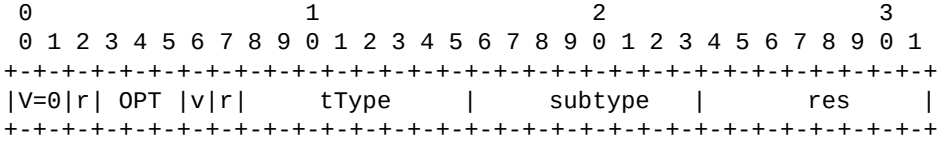
operation type (OPT) : 3 bits
     This field identifies the operation type of the ADU. It is set to 1 (change).

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

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

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

3.2.3 Change Telepointer type

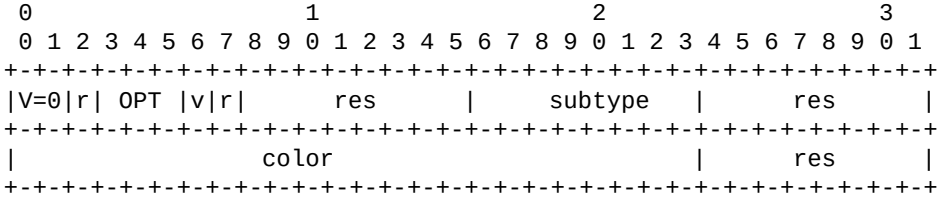


operation type (OPT) : 3 bits
     This field identifies the operation type of the ADU. It is set to 1 (change).

Telepointer type (tType) : 8 bits
     This field identifies the new appearance type of the telepointer.

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

3.2.4 Change Telepointer color

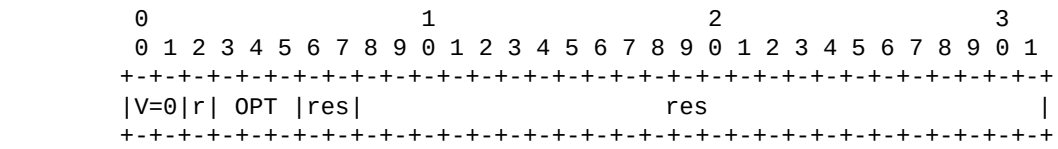


operation type (OPT) : 3 bits
     This field identifies the operation type of the ADU. It is set to 1 (change).

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

color : 24 bits
     This field identifies the new color of the telepointer.

3.2.5 Delete Telepointer ADU



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

4 Reserved Numbers

4.1 ADU Types

```
0 state
1 change
2 delete
```

4.2 Telepointer Types

```
0 east
1 north
2 west
3 south
4 north-east
5 south-east
6 south-west
7 north-west
```

4.3 Change Types

```
0 change visibility
1 move
2 change type
3 change color
```

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. Kuhmn̈ch, 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.