Request for Comments
Category: Informational

J. Muia
110856830
R. Sharma
110142530

Real-Time Collaborative Whiteboard Protocol

Abstract

The Real-Time Collaborative Whiteboard Protocol (RTCWP) is an application-level protocol designed to enable the collaborative editing of a single document, specifically drawing on a shared whiteboard. It is a stateful, persistent protocol designed for a limited set of specific tasks.

Table of Contents

1. Introduction

1.1. Purpose

The Real-Time Collaborative Whiteboard Protocol (RTCWP) is an application-level protocol designed to enable the collaborative editing of a single document, specifically drawing on a shared whiteboard. It is a stateful, persistent

protocol designed for a limited set of specific tasks.

1.2. Architecture

This protocol was designed with a single-server multiple-client architecture in mind. The server is responsible for coordinating clients and ensuring synchronization among them.

1.3. Basic Rules

OCTET = <any 8-bit sequence of data>
DIGIT = <any US-ASCII digit "0".."9">
CR = <US-ASCII CR, carriage return (13)>
LR = <US-ASCII LF, linefeed (10)>
SP = <US-ASCII SP, space (32)>
COLON = <US-ASCII colon, : (58)>
EQ = <US-ASCII equal sign, = (61)>
AMP = <US-ASCII ampersand, & (38)>
COMMA = <US-ASCII comma, , (44)>

“literal” = Quotation marks surround literal text. Unless stated otherwise, the text is case-sensitive.

rule = The character "" preceding an element indicates repetition. The full form is "<n>*<m>element" indicating at least <n> and at most <m> occurrences of element. Default values are 0 and infinity so that "*(element)" allows any number, including zero; ”1*element" requires at least one; and "1*2element" allows one or two.

(rule1 rule2) = Elements enclosed in parentheses are treated as a single element.

2. Messages

2.1. Message Format

Messages exchanged as part of this protocol are intended to be exchanged between clients and the server, but not between clients (i.e. peer-to-peer).

Messages begin with an Action token, followed by a COLON, and finally the Action-Body token. No CR or LF are allowed except in the final CRLF sequence ending the Message. See:

Message = Action COLON Action-Body CR LF

where:
Action = 1*OCTET and belonging to a predefined set of actions

Action-Body = *OCTET

3. Colour

Colour is an opaque sRGB colour represented by a 32-bit integer. The specified combined RGB value consisting of the red component in bits 16-23, the green component in bits 8-15, and the blue component in bits 0-7. Bits 24-31 are reserved for the alpha component, which is assumed to be 255 in this version of the protocol.

4. Point

A point is simply a three-tuple consisting of (x, y, colour). x and y are 32-bit signed integers representing a point in the whiteboard. Due to the representation, the protocol limits the size of the whiteboard from 0 to $2^{31}-1$ in both the x and y axis. In future versions of the protocol these values could be interpreted as unsigned in order to expand the whiteboard size.

5. Whiteboard

A whiteboard is a set of coloured points (x, y) drawn onto a finite, discrete white plane in two-dimensional space. In the intended application, the coordinate system starts at (0,0)in the top-left corner of the whiteboard and increases positively moving towards the bottom-left corner. While the actual board could theoretically be any two-dimensional shape, the coordinate system is based on the bounding rectangle of the shape of the board.

In this version of the protocol, the whiteboard is limited in size due to the constraints of the Point system described in the next section.

6. Message Definitions

For the following definitions, let INT = the 32-bit signed integer such that INT = 1*10 DIGIT

6.1. Color

A Color-message is sent from the server to the client to communicate what colour the client should be using.

Although it was in the scope of the intended application, a Color-message could also be sent from the client to the server in order to request a specific colour (without a guarantee of that request being accepted).

Color-message = “color" COLON Color-body CR LF

6.1.1. Color-body format

Color-body = INT

where the INT represents the RGB colour as described in the Colour section.

6.2. Point

A Point-message is sent from a client to the server in order to tell the server the client has drawn a new point on the whiteboard.

The same Point-message is also relayed from the server to the other clients to inform them that the point was drawn by another client.

Point-message = point COLON Point-body CR LF

6.2.1. Point-body format

Point-body = “x” EQ INT AMP “y” EQ INT AMP “rgb” EQ Color-body

where the first INT represents the x-coordinate and the second INT represents the y-coordinate.

6.3. Whiteboard

A Whiteboard-message is sent from the server to the client when a client first connects to the server. It is used to send the current state of the whiteboard such that new clients see the full picture.

Whiteboard-message = “whiteboard” COLON Whiteboard-body CR LF

6.3.1. Whiteboard-body format

Whiteboard-body = *(*(Point-body COMMA) 1*Point-body)

In simpler terms, it is a collection of Point-body elements separated by commas.

7. Synchronization

In order to achieve synchronization of clients, a simple, limited implementation of Operational Transformation is used (n.b.

alternatives such as Differential Synchronization were considered but deemed to involved provided the scope of the application). The whiteboard starts blank and has updates applied to it, which are relayed to every client.

In cases where two clients draw something at the same time, it is up to the server to decide order and relay the appropriate order of updates to every client.

For simplicity, changes are not drawn to the local (client) whiteboard until the server accepts those changes. Over a poor network connection this will hamper the user experience, however, it provides a simple approach to ensuring synchronization of clients.

8. Acknowledgments

This specification was deeply influenced by the HTTP/1.0 specification [1]. It is with due regard that Berners-Lee et al receive our acknowledgment and thanks for providing a memo that proved to be the template for this RFC.

9. References

[1] Berners-Lee, T., Fielding, R., and Frystyk, H., "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, MIT/LCS, UC Irvine, May 1996

10. Authors' Addresses

Joseph Muia
muia6830@mylaurier.ca
110856830

Robin Sharma
shar2530@mylaurier.ca
110142530