Visual Scene Update Protocol

VSUP/1.0, Draft 1

November 2024

Igni Project, Draft 1

This document is an informal draft. It may not be entirely unambiguous, grammatically correct or consistent. The document, as it stands, merely exists to assist development of a more refined document later on.

Despite the contents of this document being subject to change, the protocol itself shall remain unchanged.

Table of Contents

1 Introduction....2
2 Definitions and Abbreviations....2
3 Table Formats....3
  3.1 Data Structures....3
  3.2 Enumerations....3
4 Data Formats....3
  4.1 String....3
  4.2 File Path....3
5 Error Response....4
6 Requests....5
  6.1 Configure Connection....5
  6.2 Load Model....6
  6.3 Show Model....7
  6.4 Hide Model....7
  6.5 Transform Model....8
  6.6 Destroy Model....9
  6.7 Create Action Instance....10
  6.8 Set Action Frame....11
  6.9 Set Action Weight....11
  6.10 Destroy Action Instance....12
  6.11 Transform Viewpoint....12
  6.12 Set Viewpoint Field of View....13

1 Introduction

The Visual Scene Update Protocol (VSUP) is an application layer communication protocol that is lightweight and extensible enough to transfer modern 3D graphics over a network.

This protocol requires a bi-directional client-to-server network connection to operate.

2 Definitions and Abbreviations

3D

3-dimensional

field

variable in a data structure

float

IEEE 754 floating-point integer

ID

unique identifier

int

signed integer

shader

executable code which renders graphics

uint

unsigned integer

opcode

operation code – a unique number which identifies a command

3 Table Formats

To increase legibility and clarity, some data has been summarised as tables. The formats which these tables adhere to are detailed in the following subsections.

3.1 Data Structures

A data structure may be formatted as a table with 3 columns:

Size (bytes)	Value	Description
The size of a field	Contents of field	A brief description on the purpose of the field

Each row in the table shall outline the properties of a field in its data structure. Fields are ordered from top to bottom; The top row describing the first field, and the bottom row describing the last field.

3.2 Enumerations

An enumeration may be formatted as a 2 column table:

Value	Description
0	A brief description of the value and its properties

Each row in the table shall list a constant value alongside its description. Values shall be listed lowest to highest, with the lowest value first at the top of the table.

4 Data Formats

4.1 String

The term ‘string’ in this document specifically refers to a null-terminated string in UTF8 (RFC 3629) format.

4.2 File Path

The term ‘file path’ in this document specifically refers to a string of a UNIX-like absolute file path.

File paths shall obey the following rules:

1. The directory separator is one character, a forward slash (/).
2. A file path may be prefixed with forward slash to signify that it is an absolute path. This, however, shall not make a difference as all file paths are interpreted as absolute.

5 Error Response

Size (bytes)	Value	Description
1	8-bit uint	Request opcode
1	8-bit int	Error code

The error response shall be comprised of 2 fields. The first field, **Request opcode**, shall hold the opcode of the command which caused the error. **Error code**, the second field, shall contain the correct identifier of the error which has occurred.

Each error has a unique identifier. Error responses may use the following error codes:

Value	Description
0	Invalid protocol version
1	Model not found
2	Action not found
3	File import failed
4	ID already taken
5	Invalid opcode

6 Requests

A client may request a small change be made to its scene.

Each request has an opcode placed at its beginning. The opcode determines the actions taken by the server including how the remainder of the request shall be interpreted.

Upon an unsuccessful request, the server shall send an error response to the client responsible for the faulty request. Visually, a failed request, even one which is partially executed, shall have no effect.

A request may cause 2 or more errors at once. If this is the case, the server shall report all errors from the request one by one.

Errors

Any request shall fail if:

Invalid opcode

- The opcode of the request is not supported by this protocol.

6.1 Configure Connection

Size (bytes)	Value	Description
1	0	Opcode
4	0	Version ID

The ‘Configure Connection’ request shall adjust the properties of a clients connection with the server.

Version ID shall determine the version of protocol to use during client-server communication. To inform server that the client is communicating through version 1.0 of the VSUP protocol, **Version ID** shall be set to 0.

Errors

The ‘Configure Connection’ request shall fail if:

Invalid protocol version

- **Version ID** does not identify an existing version of the VSUP protocol.

6.2 Load Model

Size (bytes)	Value	Description
1	1	Opcode
4	32-bit int	Model ID
1+	File path	Model path

The ‘Load Model’ request shall instruct the server to import a model from a file. If the file specified by **Model path** is a symbolic link, then the file loaded shall be that which the link or chain of links point to. Files of the same base name as **Model path** may be imported and bound to the loaded model automatically.

The model created by the request must have the same ID number as **Model ID**. Upon creation, the model shall be hidden. Its initial location and rotation coordinates shall be 0 on all axes. Scale, however shall start out at 1 on model load.

All coordinates of elements within the model shall be interpreted as local to the transformation of the model.

Errors

The ‘Load Model’ request shall fail if:

File import failed

- The server fails to import the model file.

6.3 Show Model

Size (bytes)	Value	Description
1	2	Opcode
4	32-bit int	Model ID

This ‘Show Model’ request shall make the model specified by **Model ID** visible to the viewer.

Errors

The ‘Show Model’ request shall fail if:

Model not found

- **Model ID** does not match the ID of any model in the clients scene.

6.4 Hide Model

Size (bytes)	Value	Description
1	3	Opcode
4	32-bit int	Model ID

This request shall hide the model identified by **Model ID**. A model which is hidden shall not be displayed onscreen.

Errors

The ‘Hide Model’ request shall fail if:

Model not found

- **Model ID** does not match the ID of any model in the clients scene.

6.5 Transform Model

Size (bytes)	Value	Description
1	5	Opcode
4	32-bit int	Model ID
4	float	X location
4	float	Y location
4	float	Z location
4	float	X rotation
4	float	Y rotation
4	float	Z rotation
4	float	X scale
4	float	Y scale
4	float	Z scale

The ‘Transform Model’ request shall set the location, rotation and scale of the model referenced by **Model ID**. The X, Y and Z location, rotation and scale coordinates of the selected model are to be copied from the request.

Errors

The ‘Transform Model’ request shall fail if:

Model not found

- **Model ID** does not match the ID of any model in the clients scene.

6.6 Destroy Model

Size (bytes)	Value	Description
1	6	Opcode
4	32-bit int	Model ID

This request shall remove the model specified by **Model ID** from its scene.

Errors

The ‘Destroy Model’ request shall fail if:

Model not found

- **Model ID** does not match the ID of any model in the clients scene.

6.7 Create Action Instance

Size (bytes)	Value	Description
1	7	Opcode
4	32-bit int	Model ID
4	32-bit int	Action ID
1+	string	Action name

Models may contain a set of animation sequences called actions. These sequences may animate all properties of a model, including those adjustable through requests.

This request shall create a reference to an action of the same name as **Action name** within the model specified by **Model ID**. Upon creation, the action instance shall have its scale set to 0. More information on action scaling and blending is available in the **Scale Action** section.

Errors

The 'Create Action Instance' request shall fail if:

Model not found

- **Model ID** does not match the ID of any model in the clients scene.

Action not found

- The model specified by **Model ID** has no action whose name matches the string provided by **Action name**.

ID already taken

- An action of ID **Action ID** already exists in the clients scene.

6.8 Set Action Frame

Size (bytes)	**Value**	**Description**
1	9	Opcode
4	32-bit int	Action ID
4	32-bit int	Frame index

The ‘Set Action Frame’ request shall set the current frame of an action. The action to have its frame number set must be identified by same ID as **Action ID**. The **Frame index** field shall specify the frame which the selected action shall skip to.

Errors

The ‘Set Action Frame’ request shall fail if:

Action not found

- **Action ID** does not match the ID of any action in the clients scene.

6.9 Set Action Weight

Size (bytes)	**Value**	**Description**
1	10	Opcode
4	32-bit int	Action ID
4	float	Weight

The ‘Set Action Weight’ request shall adjust the blend weight of an action. The action identified by **Action ID** shall have its blend weight set to the value of the **Weight** field.

Errors

The ‘Set Action Weight’ request shall fail if:

Action not found

- **Action ID** does not match the ID of any action in the clients scene.

6.10 Destroy Action Instance

Size (bytes)	Value	Description
1	11	Opcode
4	32-bit int	Action ID

The ‘Destroy Action Instance’ request shall remove the action instance identified by **Action ID** from its scene.

Errors

The ‘Destroy Action Instance’ request shall fail if:

Action not found

- **Action ID** does not match the ID of any action in the clients scene.

6.11 Transform Viewpoint

Size (bytes)	Value	Description
1	19	Opcode
4	float	X location
4	float	Y location
4	float	Z location
4	float	X rotation
4	float	Y rotation
4	float	Z rotation

The ‘Transform Viewpoint’ request shall set the location and rotation of the viewpoint. The command shall set the X, Y and Z location and rotation coordinates of the world viewpoint to those specified by the request.

Errors

The ‘Transform Viewpoint’ request has no errors.

6.12 Set Viewpoint Field of View

Size (bytes)	Value	Description
1	20	Opcode
4	float	Field of view

The ‘Set Viewpoint Field of View’ request shall set the angular extent of the viewpoints perspective. The field of view of the viewpoint shall be set to the value of **Field of view**.

Errors

The ‘Set Viewpoint Field of View’ request has no errors.

End Draft