

CICP v1 Spec

Content Injection and Control Protocol is a system for allowing an external application called a Producer to place interactive content into a virtual world, called a World.

To implement CICP for a virtual world, you have to create three objects:

Graph

An instance of Graph is a distributed object within the virtual world and represents a single managed entity created by a Producer. A Graph contains one to many GraphNodes.

GraphNode

A GraphNode object is a single geometric or mesh shape which a Graph object adds to the scene graph; a GraphNode must respond to mouseover and mouse click events.

GraphManager

GraphManager is an object which stores references to Graph objects, keyed by graph id.

Basic Flow

CICP begins with the creation of an uninitialized Graph object in the world. This could be via configuration, via a user pulling an object from their inventory, or however the particular virtual world chooses to implement it. The uninitialized Graph object can automatically perform the "Init" handshake with a Producer, or can provide a menu allowing a user to trigger the "Init" handshake. The object must have a hostname and port of a Producer; configuration of this can be done in whatever way makes sense for the particular virtual world system.

The "Init" handshake consists of the Graph object opening a socket to the Producer and sending the line "init,<x>,<y>,<z>", where <x>, <y>, and <z> are the current world coordinates of the object. The Producer will then respond with a single line containing a "Create command". Specification of the "Create command" follows.

```
create,<x pos>,<y pos>,<z pos>,<x size>,<y size>,<z size>,<graph id>,<definition url>
```

<x pos> is the new x origin of the object in world coordinates

<y pos> is the new y origin

<z pos> is the new z origin

<x size> is the size in world coordinates of the x dimension of the Graph object

<y size> is the size of the y dimension

<z size> is the size of the z dimension

<graph id> is an integer id which uniquely identifies this Graph object to the Producer and which must be sent back to the Producer when users interact with this Graph object

<definition url> is a url to a "Graph definition file" which describes all the GraphNode objects belonging to this Graph object, and the interactive menus those objects must present to users

The Graph object must fetch the definition file and reconfigure itself according to its contents. The Graph object must also store the "graph id" to use in future communication with the Producer.

GraphNode Builtin Functionality

All GraphNodes must be able to display a text title, display a menu, and highlight themselves. Typically, the title will come up upon mouseover and remove itself when the user mouses off; upon click of a node the title will toggle on or off. Upon click of the title a menu will come up. The menu will include an option to close it and turn off the title. Menu options which specify highlighting cause the node or a group of nodes to switch to the highlight colors specified in the definition file.

Graph Object Synchronization Across All Virtual World Clients

The Graph object must synchronize itself across all clients joined to the virtual world containing it. When the "Init" handshake is performed by a given local client, it must send a message causing all clients to fetch the Graph definition file locally and reconfigure. Similarly, when a user clicks a particular GraphNode, causing its title to stay on, the title of that GraphNode should come on and stay on for all connected clients; selecting highlight for a given node or set of nodes should cause them to highlight for all clients.

Graph Definition File Format

The Graph definition file has three sections: menu definition section, node definition section, and create command section. Note that all menu titles will be URL Encoded (in the below example they are not).

Example file:

```
menu,r
-1,1,highlight
endmenu
menu,n
-2,1,reset highlights
1,1,use all for brightness
-3,1,delete
3,1,move out x
4,1,move in x
5,1,move out z
6,1,move in z
endmenu
menu,x,autohighlight
-4,1,highlight
1,1,use for brightness
3,1,use for height
2,1,sort by average
endmenu
menu,z
-4,1,highlight
3,1,use for height
```

```
2,1,sort by average
endmenu
nodes
rec,0.0,0.0,20.5,0.25,0.5,0.0,0.5,0.2,0.6,1.0,0.5,0.6,0.2,0,0,41,2005,z,1
rec,0.0,0.0,20.0,0.25,0.5,0.0,0.34,0.1,0.4,1.0,0.34,0.4,0.1,0,0,40,2004,z,2
rec,0.0,0.0,19.5,0.25,0.5,0.0,0.5,0.2,0.6,1.0,0.5,0.6,0.2,0,0,39,2003,z,3
rec,0.0,0.0,19.0,0.25,0.5,0.0,0.34,0.1,0.4,1.0,0.34,0.4,0.1,0,0,38,2002,z,4
...
create,44.0,1.0,0.0,27.0,0.0,20.5,2,http://wick.greenphosphor.com/graphs/graph_2.txt
```

Node line format:

example line:

```
rec,0.8,0.2,7.2,0.1,0.12,0.12,0.5029472,1.0,0.12,0.5029472,0.12,1.1049055,4,23,36,title,r,231
```

the names of the fields are:

```
type, xloc, yloc, zloc, xloc2, yloc2, zloc2, r, g, b, alpha,
highlightr,highlightg,highlightb,xindex, yindex, zindex, title, menuid, nodeid
```

type: can be rec (bar), rec2 (rectangular solid), sph (sphere), lin (line), or other things in the future

xloc, yloc, zloc: origin of the node in world coordinates, RELATIVE to the origin of the Graph

for line (lin): draw a line from xloc,yloc,zloc to xloc2,yloc2,zloc2

for bar (rec): create a box with base centered at xloc, yloc, zloc; square base with width of xloc2 * 2; and height of yloc2

for rectangular solid (rec2): create a rectangular solid conforming to the bounds defined by the two points xloc,yloc,zloc and xloc2,yloc2,zloc2

for sphere (sph): create a sphere centered at xloc,yloc,zloc and with radius of yloc2

r, g, b are values for red green and blue

alpha: controls transparency

highlightr, highlightg, highlightb are rgb values for the node when it is highlighted

xindex, yindex, zindex are the indices corresponding to the node's values on the x, y, and z dimensions

title is a URL Encoded text title that comes up when the node is moused over or clicked on

menuid is the id of the menu that comes up when the node's title is clicked on; an empty menu id means no menu

nodeid uniquely identifies the node and is what gets sent back to the Producer when a menu option is selected

Graph Object Builtin Functionality

A menu option id of -3 means delete. When such an option is selected by a user, the Graph object must open a socket to the Producer and send the line "<graph id>,<node id>,-3"; regardless of what is received over the socket from the Producer, the object should send a

message out to its copies on other clients indicating that it should be deleted, and then should delete itself. All copies should delete themselves as well upon receipt of the message (or however such an object deletion is supported in the given virtual world system).

A menu option of -2 means reset; all titles which are turned on must be turned off, and all highlighting must be turned off. A message must be sent out causing the same thing to happen on all copies of the object. Nothing need be sent to the Producer.

Communication to Producer Upon User Selection Of Menu Option

When a user selects a menu option which is not a built-in option (built-in options always have a negative id), the virtual world system (usually the Graph object itself, locally on the user's client) must open a socket to the Producer and send the line "<graph id>,<node id>,<option id>". The Producer will send back a new Create command or a Modify command. If the graph id contained in the new Create command is currently in use by a Graph object, that Graph object must fetch the new definition file specified in the Create command and reconfigure itself accordingly (this may be implemented as a delete of the existing Graph object and creation of a new one). If the graph id is not in use, a new Graph object must be spawned at the location contained in the Create command and must fetch the file and configure itself. If the command is Modify, then the graph object corresponding to the graph id must be modified. The file format for modifying a graph is the same as that for defining a graph, except with no menu section. The file starts with the "nodes" line and will include a line for each node to be modified. Not all nodes need be included. Modify and Create actions must take place on all connected clients.

Determining what graph id's are in use and what their corresponding Graph objects are is the purpose of the GraphManager object.

Highlight and Auto-highlight Functionality

A menu option id of -1 always means "highlight". When a user selects this option, the selected GraphNode must change its color to the rgb highlight values defined for it, and send a message causing all distributed copies to do the same. No message need be sent to the Producer.

A menu option id of -4 always means "Group Highlight". When a user selects this option, the GraphNode must check its xindex, yindex, and zindex values. If only one of them is non-zero, the Graph object should perform a highlight on every GraphNode which has the same id... for example, if xindex is 3 and yindex and zindex are 0, then every GraphNode with an xindex of 3 would be highlighted.

Auto-highlight functionality is enabled when the definition file includes a menu declaration line ending with ",autohighlight"; for example, from the definition file excerpt above, "menu,x,autohighlight". Every GraphNode which uses the "x" menu definition must then perform a Group Highlight upon mouseover, LOCALLY ONLY (only on the client of the user who moused over).

Inactive Command From Producer

Instead of returning a new "create" command to the virtual world via a socket, a Producer

may return the line "inactive". Upon receipt of this line, the Graph which the user interacted with should deactivate its menus and modify its title to indicate to the user that it is no longer active. The Producer will use this option when it receives a message from a Graph with a graph id that it does not recognize. The inactive Graph must still allow the user to delete it.

Multi-select Menu Options

The second parameter in a menu option definition line within a definition file can be either "1" or "2". If it is "1", the message sent to the Producer upon selection of that menu option has the format "<graph id>,<node id>,<option id>"; if it is "2" the message sent to the Producer has the format "<graph id>,<node id>,<option id>,<list of selected nodes>", where <list of selected nodes> is a semicolon-delimited list of the id's of the GraphNode objects within the Graph that have their titles clicked on.

Example menu option definition line:

```
1,2,drill massspec_patterns
```

Clicking on a node that uses this menu and selecting this option might result in the following being sent to the Producer, assuming three nodes have their titles clicked on:

```
3,4,5,2;4;7
```