Our Little Multiuser Dungeon (OLMUD)

Table of Contents

1 SrrTrains is Dead, Long Live SrrTrains!!!

Dear Reader,

Well, I had invested a lot of sweat into the SrrTrains project (between the years 2008 and 2019), in the end it has been frozen somehow.

Nevertheless, I can still remember a lot of the findings that I was granted during that project, so now it makes totally sense, when I try to write down my thoughts about John's projects.

John is currently doing his JSONverse, which is – as far as I understood so far – a WebGL based, X3D based experimental implementation of some Network Sensor Prototype (NSP) and its application within one (or more than one) exemplary multiuser scene(s).

You can find the JSONverse at following link: https://github.com/coderextreme/JSONverse.

1.1 Scope of this Hobby Report

This Hobby Report cannot be an official documentation of the JSONverse (JV), but it sketches a few concepts that have been, shall be, should be, might be or can be implemented in some past, present or future development step of the JV.

I cannot be more specific, because I am not a member of the JV, I am just a friend.

However, the first demo scene of John's JSONverse has been installed on my vServer and John is trying to keep it up to date at following address:

https://lc-soc-lc.at:8443/jsonverse/apache.html

This address is protected by a simple password, to mitigate DDOS attacks

u=doug

p=freewrl

1.2 Restrictions of the Demo Scene

As written above, the demo scene (DS) is maintained at following address:

https://lc-soc-lc.at:8443/jsonverse/apache.html

This address is protected by a simple password, to mitigate DDOS attacks

u=doug

p=freewrl

It might be obvious to the reader, that the DS cannot support (yet) all the concepts that are described in the present Hobby Report, and it might support additional concepts that are not (yet) described in the present Hobby Report.

Such discrepancies, as far as I am aware of them, are marked in a grey block, like the following.

DS Restriction: The demo scene does not (yet) support ALL concepts that are described in the present Hobby Report, but it might support additional concepts that are not (yet) described here.

2 Multiplexing

Before we dive a little bit into the needed structure of such multiuser scenes (what I called SMS, some time ago), we need to talk about the technical provisions that are provided by the Internet.

Often, we refer to the Internet as a "Network of Computers", but actually it is a "Network of Processes", as I will elaborate shortly.

The Internet addresses, the @IP, are sometimes referred to as revealing your identity.

Well, that's not completely true. Actually, an @IP reveals the identity of a machine. Only, if it is somehow proven that you are using (have been using) this machine, then the @IP reveals your identity, too.

Second, a machine, sometimes referred to as a host, hosts one or more processes. Now, it's the processes that actually run the software of your computer. We can think of the processes as of little busy dwarves that are inhabiting the machine.

Now, if one of these dwarves, who inhabits one machine, wants to send a letter to a dwarf, who inhabits another machine, he needs to ADDRESS the dwarf.

Like in real mail, where the address consists of NAME + ADDRESS of a person, when sending an Internet DATAGRAM to another dwarf, the address consists of PORT + @IP.

Therefore, the first dwarf, let's call him "Dwarf A" creates a connection to "Dwarf B", and then the datagram is sent via the connection.

Hence, such a connection is identified by a quadruple:

Network Connection is identified by: @IP A + PORT A + @IP B + PORT B

as depicted in following Abbildung 2.1.

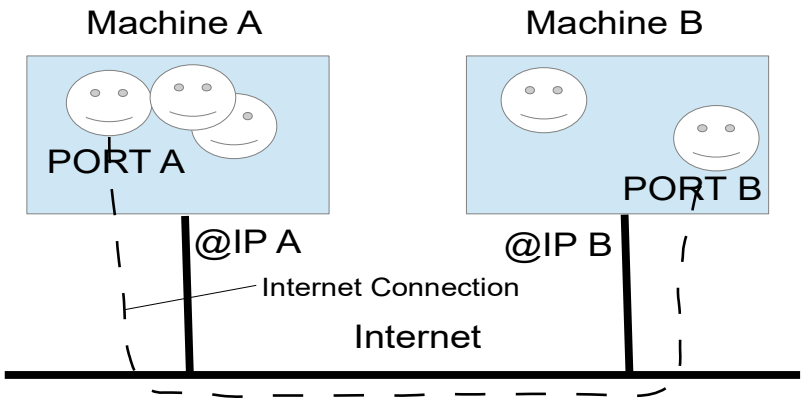

Abbildung 2.1: Basic Multiplexing in the Internet

So, this figure depicts the communication among processes in the Internet, where always two processes are connected by an Internet Connection.

3 Additional Multiplexing (Sub-Multiplexing)

However, when we talk about X3D scenes, then the processes – **what we call the Client Applications** – do not provide sufficient granularity for the network communication.

When a JV Client Application communicates with an X3D Collaboration Server – or even directly with another JV Client Application –, then we have to consider **the fact that the scenes will consist of what-i-call "objects"** (in the SrrTrains project, I called such objects the "facilities").

So, although there is only one Internet Connection for each client application (i.e. for each human user) between each client application and the X3D Collaboration Server, **we have to multiplex traffic for several (or many) objects over each of those connections**.

Additionally, **we have some session level control messages (e.g. for chat)**.

In John's implementation, the multiplexing is implemented by what is called "namespaces" of the socket.io software. There we use

- The default namespace for session level control: /
- One namespace for each object of the scene: /obj1, /obj2, /obj3, ...

This is depicted in the following figure.

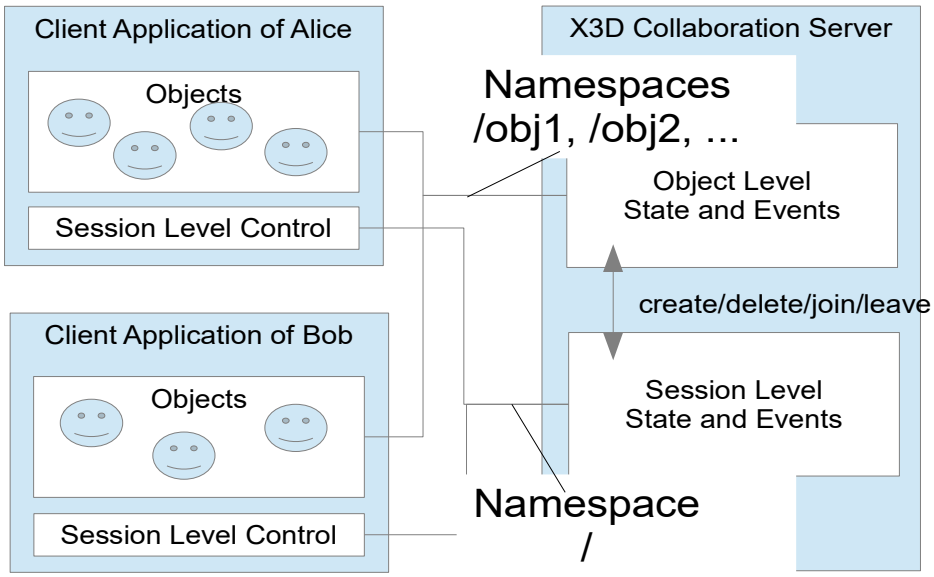

Abbildung 3.1: sub-multiplexing per namespace for objects and session level

Now, the socket.io library needs a global identifier, and additionally the server can handle more than one session and additionally each session is provided by a user of the server.

Hence the namespaces might actually look like the following:

- **Session Level Namespace: /socket.io/<user>/<sessionid>/**
- **Object Level Namespace: /socket.io/<user>/<sessionid>/<objId>**

4 The Objects of a Multiuser Scene – Shared State

The term object is one of the most general terms that we have in information technology and information science. So, what do we mean with the term object in the context of a multiuser scene?

Well, an object can be any part of an X3D scene, as long as it needs to share some information among its instances in the different client applications, on behalf of that part of the X3D scene.

This can be a model (which can be rendered), e.g. a multiuser capable slider as in the demo scene (DS), it can even be a rather complex model, like a locomotive in a trains game, or it can be an avatar[1].

Also, it can be a non-rendered part of the scene, e.g. an X3D <Script> node that shares some information among the client applications, while it calculates values for a simulation that are input for other, rendered objects.

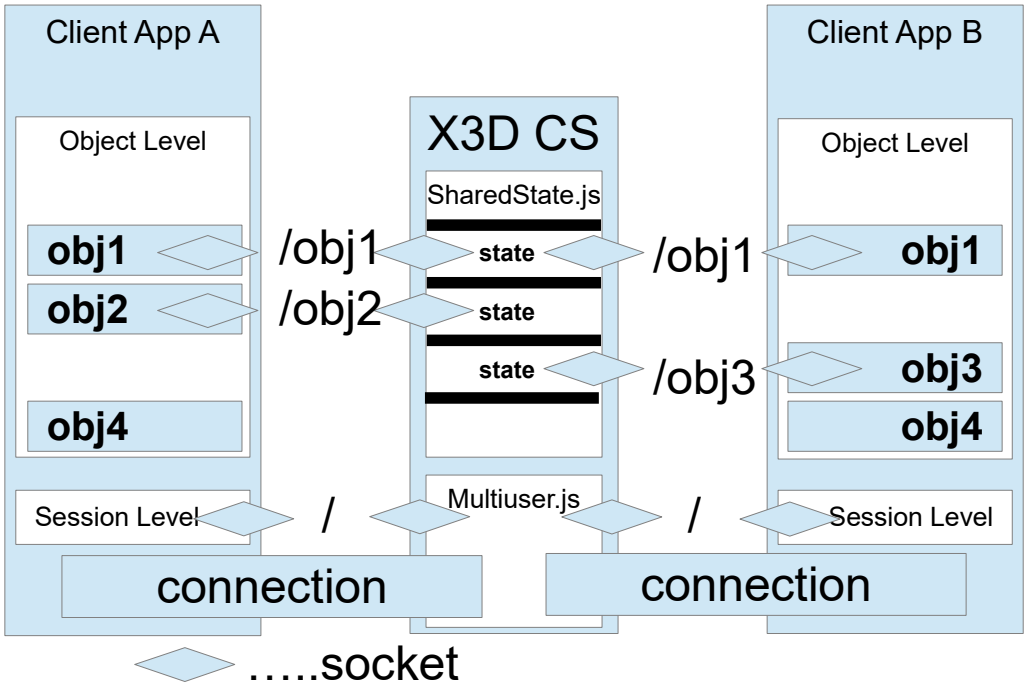

Abbildung 4.1: Example – one session with four objects

As far, as the JSONverse is involved, an object is defined by an 1:1 relation to an instance of a network sensor, i.e. by an 1:1 relationship to a socket.io socket and its namespace.

In the above example, obj1, obj2 and obj3 use the namespaces /obj1, /obj2 and /obj3 for the communication with the X3D Collaboration Server.

Acoording to our definition, obj4 is not an object, actually, because it does not share any information (states or events) among its instances.

1 An avatar is a very specific model, because it has a close relationship to a user of the multiuser session. It REPRESENTS a virtual identity of a user.

5 Actors

Tbd.

5.1 Scene Author

Tbd.

5.2 Model Author

Tbd.

5.3 Session Operator

Tbd.

5.4 Session Participant

Tbd.

6 Runtime Procedures

Tbd.

6.1 Bootstrapping the Scene / Session Description

6.1.1 Group Management

Tbd.

6.1.2 Session Description

Tbd.

6.1.3 Bootstrapping the Scene Instance

Tbd.

6.2 Login and Avatar Selection

Tbd.

6.3 Event Distribution

Tbd.

6.4 Shared State

Tbd.

6.5 Server Side Calculations

Tbd.

6.6 Flexible Client Side Calculations by the Author

Tbd.

7 Used Terms and Abbreviations

This Hobby Report has used following terms:

JSONverse (JV) The JSONverse is a collection of HTML/CSS/JavaScript/X3D software by John C., that aims for the experimental support of various types of 3D multiuser sessions, based on some prototype of a network sensor.
Following 3rd party software is used: socket.io as basis for the network sensor and other network communication with the collaboration server, node.js is used as platform for the server software.
The goal is collaboration within closed groups of users. Those groups of users might be small. Global collaboration is not the goal of this project.

Demo Scene (DS) The Demo Scene comes with the JSONverse. It represents the current state of implementation of the JSONverse.

Management App The JSONverse comes with a management app, which allows a managing user to manage JV groups, JV sessions and JV rooms.

Managing User A managing user is a user, who uses the management app in order to create, distribute and withdraw group tokens, as well as session tokens. Also, the management app supports the managing user with the creation, update and distribution of Session Descriptions.

Session Description (SD) A Session Description is a (JSON) data object that describes a session. I would strive for compatibility with RFC 8866, however the current implementation of the JSONverse needs more information than is standardized in RFC 8866.

Client Application Some HTML/CSS/JavaScript/X3D application that implements the UI of the multiuser session for one user (participant).

Object An object can be any part of an X3D scene, as long as it needs to share some information among its instances in the different client applications, on behalf of that part of the X3D scene