Simulated Railroad Framework, http://simulrr.sourceforge.net
Synopsis: 000_Synopsis

This file valid for step 0033.10
Issue Date: 2017-03-17

Glossary and Abbreviations

Synopsis

The glossary should provide a fast impression about the concepts SrrTrains and SMS, as referenced by paper 000_Synopsis.

Let's start with DIGITS – Distributed Internet Geographic Information Transmission Service

That idea suggested to store a three dimensional model of the world in a distributed data base in the Internet. The DNS (Domain Name System) of the Internet should be used to access this distributed data base.

E.g., if a company provided a DIGITS, then they should define a "DIGITS Root Server", e.g. digits.foo.com (called <root> in this text).

If an application liked to receive a three dimensional model of a part of the world, then the application would define three keys to access the data base,

1) a Virtual Roaming Area (VRA),
2) a Level of Detail (LoD) and
3) a Reality Identifier.

The first two parts of the key would be used to look up the address of a specific DIGITS Server in the DIGITS Root Server.

E.g., if the VRA identified an area within the 19th district of Vienna and if the LoD was "pedestrian", then the DIGITS Root Server would return the address of server krim.doebling.vienna.austria.europe.<root>.

Now the application would send a "DIGITS Request" to that DIGITS Server.

The request would traverse all servers from

- krim.doebling.vienna.austria.europe.<root> via
- doebling.vienna.austria.europe.<root> via
- vienna.austria.europe.<root> via
- austria.europe.<root> via
- europe.<root> to
- <root>,

where the "lower" DIGITS Servers would add smaller objects and the "higher" DIGITS Servers would add bigger objects, finally the <root> server would terminate the request and everything (the whole scenery, i.e. "geographic infrastructure" - GeoIS) would be sent back to the application.

Obviously some standardized data format would be necessary to transmit those data, where X3D comes to ones mind.

The second idea was SIMUL-RR – Simulated Railroad

That idea suggested to implement 3D multiuser games (scenes) to be used for training purposes at Railway Operator's premises.
Dual use could be defined to use the software for do-it-yourself-virtual-multiplayer-model-railroads in addition, in order to get synergies.
This idea led to the hobby project SrrTrains v0.01, which is now dealing with the concepts SrrTrains and SMS.

The idea SMUOS/C3P is additionally mentioned in the appendix of the present paper.

Overview

The glossary is split into three chapters.

The first chapter lists the terms and abbreviations shortly, the second chapter explains the terms in more detail.

Terms that are explained in more detail in chapter 2, are marked with an asterisk in chapter 1 (*).

Sub chapters 1.2. and 2.2. contain concepts that are definitively part of the SMS concept, sub chapters 1.3. and 2.3. contain concepts of the experimental software that might become part of the SMS concept and the paper 100_SrrFramework contains concepts of the experimental software that might not become part of the SMS concept.

The third chapter contains some basic thoughts for a use case analysis.

Appendix A contains the terms and abbreviations about the SMUOS/C3P idea.

1. List of Terms and Abbreviations

1.1. General Terms and Abbreviations

DIGITS.....	Distributed Internet Geographic Information Transmission Service (the first idea)
SIMUL-RR.....	Simulated Railroad (the second idea)
SrrTrains v0.01.....	Simulated Railroad Trains, the hobby project
Master Concepts.....*	
SrrTrains.....*	Simulated Railroad Trains, the concept
simulrr.....*	Simulated Railroad Framework, the sourceforge project
smuos.....*	Simple Multiuser Online Scenes, the sourceforge project
SMUOS/C3P.....	a future idea (X3D Component / Collaborative 3D Profile)
SMS.....*	Simple Multiuser Scenes, the overall concept

1.2. Terms and Abbreviations about SMS (Overall Concept) – Not Yet Implemented!!

Reality, Virtual Reality, Real Reality.....*	
User.....*	
PSI.....*	Personal Scene Instance
SMS.....*	Simple Multiuser Scene
Multiuser Session....*	
SCSI.....*	Server/Controller Scene Instance
VLF.....*	Virtual Life Facility
RLF.....*	Real Life Facility
OM.....*	Operational Mode
RLO.....*	Real Life Object
VLA, RLA.....*	Virtual Life Avatar / Real Life Avatar
CE.....*	Collateral Entity
UPS.....*	Universal Positioning System
GeoIS.....*	Geographic Infrastructure
SYNC.....*	Synchronisation
POI.....*	Point of Interaction / Point of Interest

1.3. High Level Concepts of the Experimental Software ("smuos" and "simulrr")

```
Architecture.........*
Models...............*
MIDAS Objects........* Multiuser Interactivity Driven Animation and Simulation
                      Objects
OBCO.................* Object Controller
Modules..............*
MC...................* Module Coordinator
MOC..................* Module Controller
Frame................*
SSC..................* Simple Scene Controller
CC...................* Central Controller/Communication Controller
commState............  Communication State
CSCR.................  commState Change Request
Unbound Models.......* (not yet implemented!)
UOC..................* Universal Object Class (not yet implemented!)
Model State
Handover.............* (not yet implemented!)
Moving Modules.......* (not yet implemented!)
```

2. Description of Some Terms

2.1. General Terms

x) Master Concepts

The concept SrrTrains is built by only a few "master concepts", everything within SrrTrains is derived from these master concepts:

- build multiplayer virtual model railroads based on VRML/X3D
- make it an open concept where many people can contribute their ideas and sweat
- use open source software as far as useful and possible
- players can attain so-called "roles" (engineer, shunter, ...)
- players can communicate during the game (chat, voice-chat, ...)
- a layout and its models can be influenced by a command line interface
- authors can (easily) build models of rail vehicles
- authors can (easily) build models of houses and other static objects
- layouts are built by modules (from different authors)
- mechanisms exist to restrict usage of your models and modules

x) Simulated Railroad Trains (SrrTrains)

SrrTrains is rather a concept than a specific software that could be bought somewhere.

Sometimes, we use the term SrrTrains to explicitely specify, we build on the so-called "Master Concepts" (see above), plus the fact that we decided to use the Network Sensor node of the Web3D consortium

The word "SrrTrains" is also used as a description of the hobby project "SrrTrains v0.01", which was located at http://simulrr.wordpress.com (meanwhile deleted) and which is now hibernating at http://letztersein.wordpress.com/srrtrains-v0-01/.

x) simulrr

In year 2010 we decided to outsource the core part of the SrrTrains software to an own sourceforge project.

Now the sourceforge project "simulrr" deals with the SRR Framework, whereas the SrrTools (closed source) are maintained by the hobby project SrrTrains v0.01.

Since currently the whole story is a "one-man-show", this doesn't make a big difference.

x) smuos

In year 2013 we decided to outsource the core part of the SRR Framework to an own sourceforge project. We called that software "SMUOS Framework".

It's a future option to "go professional" with the SMUOS Framework (see the chapter "Terms of the SMUOS/C3P Idea", too). This would mean:

a) enhance the network sensor and rebase the SMUOS Framework to the enhanced network sensor
b) derive a new X3D component from the SMUOS Framework. This would finally make the SMUOS Framework obsolete

As a network sensor makes only sense, when the used communication protocol is standardized, this is true for a new X3D component "SMUOS", too.

x) Simple Multiuser Scenes, the Overall Concept (SMS)

If the SMUOS/C3P idea was realized, i.e. if a new component of X3D was defined and if the communication protocols were defined for the network interface, then we could think about an overall concept of SMS (Simple Multiuser Scenes), which would not be restricted to pure declarative environments.

Hence we could think about the general implications of being multiuser capable for the architecture of a scene, independently whether it is a declarative scene or not.

x) A historic note: the "simulrr" project is older than the "smuos" project. Actually the "smuos" project is a simplification of the "simulrr" project – just omitting the railway stuff.

The SMUOS Framework was created by taking the former "base module" of the SRR Framework (while "step 0033.08" of the SrrTrains v0.01 project was already hibernating) and by doing some renaming.

Afterwards, in "step 0033.08" the SRR Framework replaced its "base module" by the SMUOS Framework, now having a kind of "equation" (valid from "step 0033.08" onwards):

SRR Framework = SMUOS Framework + Train Manager Extension

2.2. Terms about the Overall Concept SMS (Not yet implemented!!!!!)

x) Reality, Virtual Reality, Real Reality

There is only one *reality*, but every person carries an own model of the reality in his mind.

This model helps the person to foresee the future development of reality and it helps the person to influence reality according to his will.

Virtual reality is a part of the reality that is implemented by means of technology and that helps one or more persons (see user) to inhabit a virtual scene that needs not be directly related to the reality.

Strictly spoken, an ancient form of virtual reality is already to sit around the camp fire telling stories. Also books and movies form kinds of virtual reality.

Usually we use the *narrow term* virtual reality, if some minimum technological requirements are fulfilled, e.g. the usage of stereoscopic computer graphics.

We use the term *real reality* to denote all parts of the reality that are not part of the virtual reality in question, but that are of relevance for the virtual reality.

Anything else is just reality.

x) User

A user is a person who uses a personal scene instance (see below) to inhabit a Simple Multiuser Scene (see below) in the course of a multiuser session (see below).

x) Personal Scene Instance (PSI)

A personal scene instance is the collection of all technological facilities that are needed so that one user can inhabit a Simple Multiuser Scene.

One important facility of the PSI can be a Web3D browser that interprets a concrete scene graph.

The user interface of the PSI can be used via the senses and skills (SaSk) of the user.

x) Simple Multiuser Scene (SMS)

A Simple Multiuser Scene is a collection of facilities that are accessible via standardized protocols and that can be instantiated within PSIs to provide virtual senses and skills (vSaSk) to users.

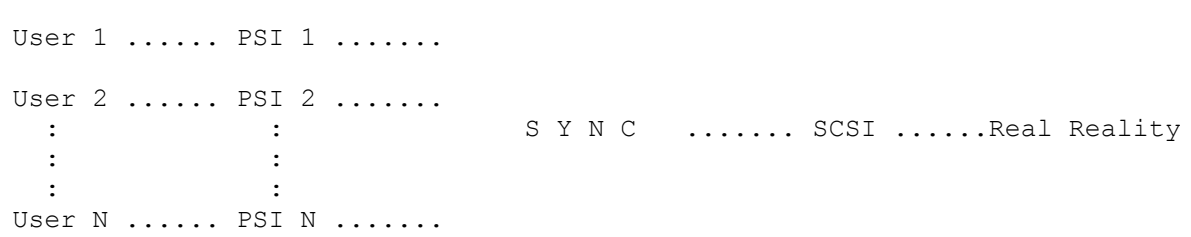
Such facilities include, e.g. (see below for detailed definitions):
- Avatars to be able to represent virtual identies
- Models to be able to render the renderable objects of the scene
- Modules to be able to render surroundings of the scene
- Geographic infrastructure to be able to render surroundings of the scene

x) Multiuser Session

A multiuser session is an instantiation of an SMS for a concrete set of users.

Those users will be able to inhabit the virtual scene together.

Technically spoken, a multiuser session is a collection of one or more PSIs and of one optional SCSI (see below), all of which are synchronized to each other.



x) Server/Controller Scene Instance (SCSI)

The Server/Controller Scene Instance connects the multiuser session to the real reality in order to synchronize real life facilities (see below) with virtual life facilities (see below).

This enables the mixed reality mode to be used as operational mode (see below).

x) Virtual Life Facility (VLF)

Virtual life facilities are used to provide virtual senses and skills to a user. In mixed reality mode VLFs may be synchronized to real life facilities (see below).

A VLF is an instantiation of a facility of the SMS.

Examples of VLFs are:
- Virtual life avatars (or simply avatars)
  to represent virtual identities to one user
- Models
  to render the renderable objects of the scene to one user
- Modules
  to render the surroundings of the scene to one user
- Geographic infrastructure
  to render the surroundings of the scene to one user

x) Real Life Facility (RLF)

Real life facilities are parts of the real reality.

We distinguish following kinds of RLFs:
- real life avatars (see below),
- real life objects (see below) and
- collateral entities (see below).

x) Operational Modes (OM)

A multiuser session can operate in one of following modes:
- Single user mode – only one PSI exists, SCSI does not exist
- Multi user mode – more than one PSI exist, SCSI does not exist
- Mixed reality mode – at least one PSI exists, SCSI exists

x) Model, Real Life Object (RLO)

A model is an object within an SMS that can be rendered.

In other words, it is an object to the virtual senses and skills of the user, when he inhabits the SMS through the PSI.

In mixed reality mode, a model may represent a real life object (RLO).

An RLO is always represented by a model, otherwise it would be a collateral entity.

x) Avatar, Virtual Life Avatar (VLA), Real Life Avatar (RLA)

An avatar is an object that represents a virtual identity (see below). A virtual life avatar is a model that represents a virtual identity and a real life avatar is an RLO that represents a virtual identity.

x) Collateral Entity (CE)

A collateral entity is an RLF that is not an RLO. I.e. it is a real life facility that is somehow relevant for the multiuser session, but it is not modelled in the SMS.

x) Module, Universal Positioning System (UPS)

According to the MMF paradigm, an SMS consists of one or more modules that build the surroundings of the scene, whereas each renderable object (each model) is assigned to one of the modules.

A module spans a local (pseudo-) euclidean spacetime, which is used to position the models.

In mixed reality mode, we will often use WGS84 coordinates as global coordinates, which can be used to position the modules.

Hence a local coordinate system in real reality can be defined relative to GPS.

Now the concept SMS aims to be a concept for the 21st century and hence a GPS will not be enough. We will need something that includes the universe into its concepts, not only the globe.

UPS the right wording for such idea.

And it need to be hierarchical, according to the eMMF paradigm. One level of modules being the top level (within a scene) containing top level models. Each top level model may contain second level modules containing second level models and so on.

Clear, there is nothing like a „top" level in universe (in UPS), Hence the top level must be identified by gravitational field instead of velocity and position. This is ffs.

x) Geographic Infrastructure, Tiles

The relations among modules, geographic infrastructure and tiles are ffs.

x) Identity, Virtual Identity, Real Identity

Need not be defined. If we need to explain this, then we do really have a problem.

x) Synchronization

The projects "simulrr" and "smuos" use the Network Sensor / Event Stream Sensor for synchronization of scene instances. This may lead to the specification of standardized communication protocols for SMS.

The software objects that are used within PSIs and within the SCSI to synchronize the multiuser session, are called MIDAS Objects (Multiuser Interactivity Driven Animation and Simulation Objects). In case of the projects "simulrr" and "smuos" the MIDAS Objects are realized by X3D prototypes written in X3D and ecmascript.

x) Point of Interaction, Point of Interest (POI)

A POI is a unit that can be addressed within the IoT.

A Point of Interest delivers a stream of events to the multiuser session. This stream describes (a part of) the state of one or more RLOs.

A Point of Interaction accepts a stream of events from the multiuser session. This stream influences (a part of) the state of one or more RLOs.

A Point of Interaction may deliver a stream of events to the multiuser session. This stream describes (a part of) the state of one or more RLOs.

2.3. High Level Concepts of the Experimental Software ("smuos" and "simulrr")

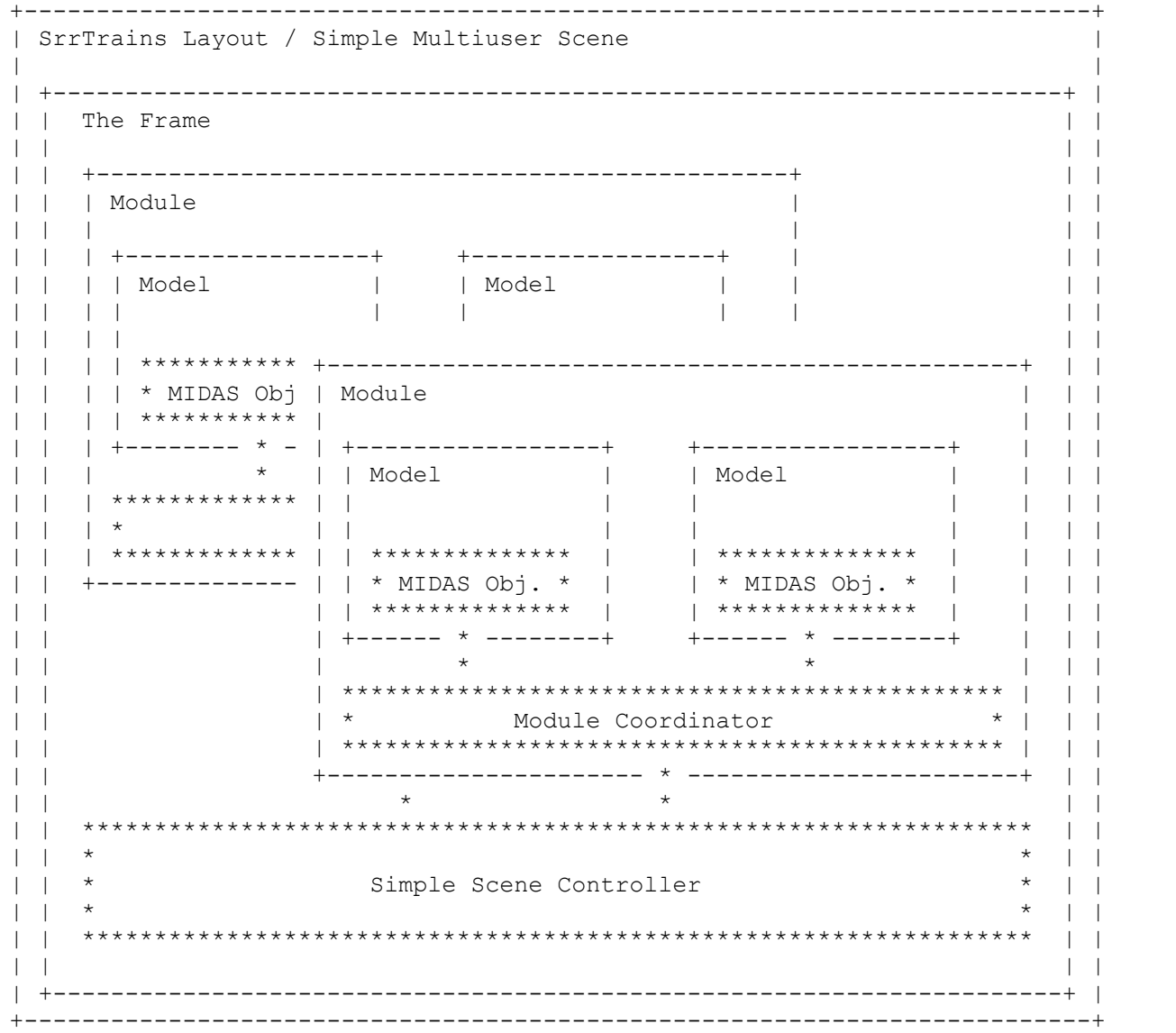
x) Architecture (MMF Paradigm)

An SrrTrains layout/Simple Multiuser Scene consists of several parts:

Models.....interactive, animated, simulated MU capable VR/AR objects that may render real life objects
Modules.....parts of the scenery that are inhabited by models; each module establishes an own local coordinate system and can be loaded/unloaded independently from other modules
Frame.....some general X3D/ECMAScript code and/or external application/web page that supports the modules and models in presenting an SrrTrains layout or Simple Multiuser Scene

The SRR/SMUOS Framework has got a set of X3D prototypes for each of these parts. Thus the SRR/SMUOS Framework can be used to build interoperable models, modules and frames.

Scene Part	Associated Part of the SRR/SMUOS Framework
Models	MIDAS Objects / SRR Objects
Modules	Module Coordinator (Base or Extension)
Frame	Simple Scene Controller (Base or Extension)



x) Models

SMS Models are interactive, animated, simulated MU capable VR/AR objects that may render real life objects.
Usually they will be provided by model authors and they will use MIDAS Objects to instrument their animation, interactivity and simulation capabilities.

x) Multiuser Interactivity Driven Animation and Simulation (MIDAS) Objects

MIDAS Objects are the parts of the SRR/SMUOS Framework that are used in models to implement multi user capable animation and interactivity (as well as simulation).
One can state, the MIDAS Objects are "invisible engines" of SMS Models.
MIDAS Objects integrate seamlessly into the animation and interactivity paradigm of VRML/X3D.

x) Object Controller (OBCO)

Many MIDAS Objects maintain global states and share those global states among all scene instances of a multiuser session.
Only one of the scene instances can attain the controller role for each one of those MIDAS Objects.
A scene instance that attains the controller role for a MIDAS Object, is said to have the OBCO role (object controller role) for this MIDAS Object.

x) Modules

SMS Modules are parts of the scenery that establish an own coordinate system each and that can be loaded/unloaded independently of each other. An SMS Module contains one instance of the Module Coordinator.

x) Module Coordinator (MC)

The Module Coordinator is the part of the SRR/SMUOS Framework that is used by the module author to coordinate the SRR/SMUOS Framework within one module. The Module Coordinator forwards information between the MIDAS Objects and the Simple Scene Controller and it provides an interface to the module author, where he can request activation or deactivation of the module. Resulting module activity will be reported at this interface, too.

x) Module Controller (MOC)

The MOC of a module is the scene instance that attains the MOC role for this module. The OBCO roles of all MIDAS Objects of this module will be assigned to this scene instance.

The central controller contains a mechanism to assign MOC roles to the scene instances depending on module activity.

It's not important for the user, which scene instances have MOC roles, but for tracing purposes the SSC provides an interface, via which the frame can request MOC roles for a scene instance.

x) Frame

Frames may be very different. They may adopt external applications or web pages to present a GUI to the user, they may provide chat or voice-chat capabilities, all of which are relatively independent from the SRR/SMUOS Framework.
SrrTrains/SMUOS only defines that each scene instance instantiates (parts of) the frame exactly once and each of these frame instances contains exactly one instance of the Simple Scene Controller.

x) Simple Scene Controller (SSC)

The SSC is the part of the SRR/SMUOS Framework that is used by the frame author to coordinate the SRR/SMUOS Framework for one layout/SMS.

The Simple Scene Controller provides an interface to
- initialize the Framework in multi-user-mode or in single-user-mode
- add/remove avatars
- report avatar position/orientation to the frame in relative coordinates
- request the controller role
- register/deregister modules
- activate/deactivate modules
- request MOC roles
- set trace levels and output tracer output
- display carried keys **)
- put carried keys into key containers or locks **)
- reset all keys **)
- send/receive console commands/responses
- display all beamer destinations *)
- bind one beamer destination *)
- join a remote user
- output short status messages

*) realized by example SSC extension "Beamer Manager"
**) realized by example SSC extension "Key Manager"

x) Central Controller (CC)

The Simple Scene Controller and the Simple Scene Controller Extensions contain client software and server software.
The client software is active in each and every scene instance of a multi-user session.
On the other hand, the server software is only active in one scene instance of all scene instances of a multiuser session.

This scene instance is said to have the central controller role.

It's not important for the user, which scene instance has the central controller role, but for tracing purposes the SSC provides an interface, via which the frame can request the central controller role for a scene instance.

x) Communication Controller

The Central Controller of the SSC Base is sometimes denoted "Communi-cation Controller", because it maintains the Communication State.

x) Unbound Models (*not yet implemented*)

Many models are loaded and initialized together with their parent modules. However, some models – unbound models – can be loaded and unloaded on demand and hence initialized during the run of the game.

x) Universal Object Class (UOC) (*not yet implemented*)

In some future release of the SRR Framework, it will be possible that unbound models will change their being attached to a module.
<ROUTE>ing the module parameters modParam of a new module to the external interface of the model will cause the model to be attached to the new module.
Hence the unbound models cannot be identified by module name + object ID (as bound models are), but they must be identified by UOC name + object ID.
Each SMUOS Extension will have the choice to support zero or more universal object classes for unbound models.

x) Handover (*future idea*)

The process of assigning/attaching an unbound model to a new module is called "Handover".
The extended object ID of the unbound model (UOC name + object ID) does NOT change during handover.

x) Moving Modules (*future idea*)

The "original" MMF paradigm defines the architecture of an SMS as
frame --- 1:n --- modules --- 1:n --- models.
Now the idea is to allow modules being parts of a model. Hence a module can be contained in a model can be contained in a module can be contained in a model and so on:
frame -- 1:n -- modules -- 1:n -- models -- 1:n modules -- 1:n -- models
Hence if a model would be a moving model, then the contained modules were moving, too.

This would establish an enhanced MMF paradigm (eMMF paradigm).

3. The System and the Actors

When one starts a use case analysis, usually one defines a "system" and some "actors".

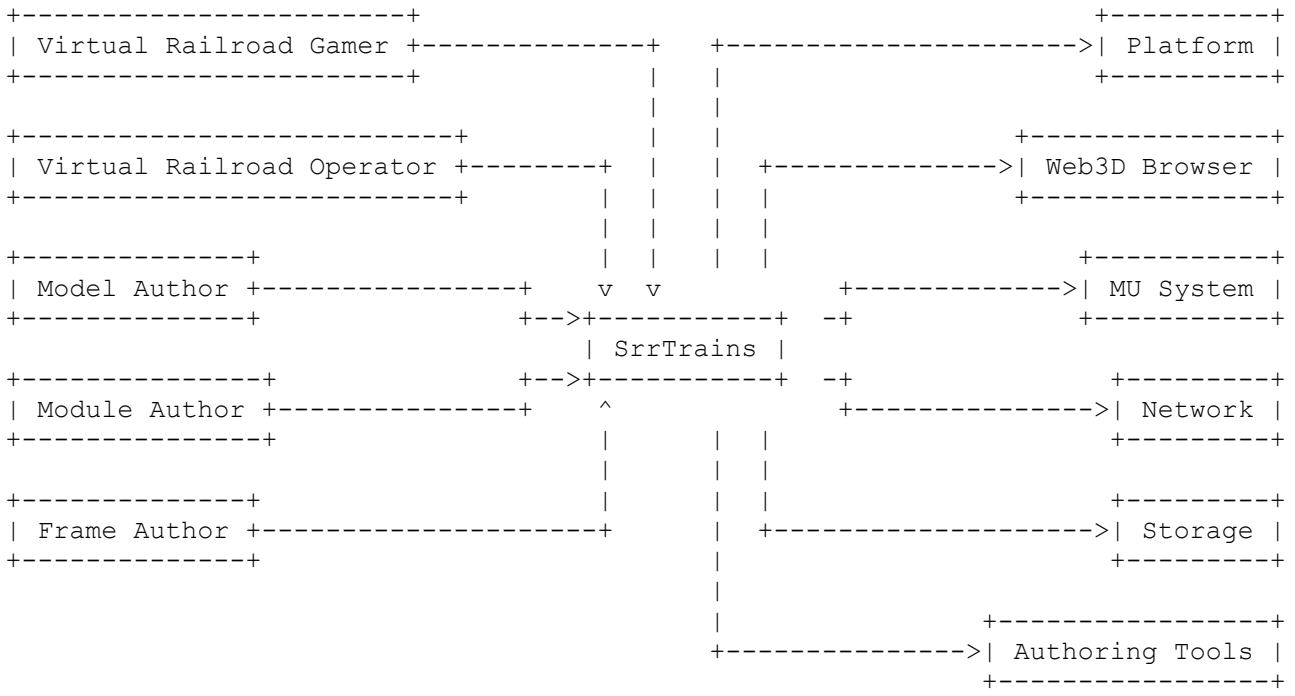
Some actors "use" the system, other actors "are used" by the system.

Our "system" is SrrTrains. SrrTrains is rather a concept than a specific software that can be bought somewhere.

By the way: the first intended use case of SrrTrains was the "Do-it-yourself-virtual-multiplayer-model-railroad". The first addresses that have dealt with this use case are

the home page "Home of SrrTrains"
http://members.chello.at/christoph.valentin
and the blog "SrrTrains's Blog"
http://simulrr.wordpress.com (meanwhile deleted)

SrrTrains is related to the following actors (usage "from left to right")



Platform: a machine, usually a PC, with some operating system and some
-------- web browser. Could also be a CAVE installation.

Web3D Browser: could be interpreted as a part of the "platform",
------------- e.g. BS Contact, Instant Player, ...

MU System: Some server to connect the network sensors of the scene in-
--------- stances. E.g. BS Collaborate, Blaxxun, DeepMatrixIP, ...

Network: some means to connect the scene instances (Web3D browsers)
------- with the MU system on IP layer (e.g. the Internet)

Storage: Some place to hold the .x3d, .jpg, .png, ...
------- files of the layout (e.g. local disk drive or
web space)

Authoring Tools: Tools, that support the authors in creating models,
--------------- modules and frames (e.g. X3D-Edit, Blender, ...)

Appendix A

A.1. Terms of the SMUOS/C3P Idea

SMUOS.....* Simple Multiuser Online Scenes, the X3D component
C3P.....* Collaborative 3D Profile
IP..... Internet Protocol
MU System..... Multiuser System
CP..... Connectivity Platform (conceptual name for an evolved CS)
SVR mode..... Single User Virtual Reality Mode
MVR mode..... Multiuser Virtual Reality Mode
SI..... Scene Instance
MR mode..... Mixed Reality Mode
SI=0..... SI with sessionId=0 (some "real" reality)
SI<>0..... SI with sessionId <> 0 (some "virtual" reality)
AC/DC modes..... Autonomous Mode – Connected / Disconnected Mode
CM..... Connected Mode
ITR..... Interface to Reality
POI..... Point of Interest / Point of Interaction

x)Simple Multiuser Online Scenes (the X3D component SMUOS)
Basically, the idea of SMUOS/C3P is to move the SMUOS Framework one layer downwards, into the Web3D Browser. Hence, the X3D prototypes could be used as a basis to define a new component of the X3D standard, the suggested name of this component being "SMUOS – Simple Multiuser Online Scenes".
Basically, this approach would give us the chance to "start from the scratch" with a completely standardized MU System, based on collaboration servers and network sensors.
This makes only sense, when the communication protocol to be used becomes a standardized protocol (see "C3P").

x)Collaborative 3D Profile (C3P)
C3P is a conceptual name for a (set of) protocol(s) that have to be used between scene instances and collaboration servers, when using the SMUOS component.
The protocol(s) need not be new protocols, maybe some rules "how to use existing protocols" may be sufficient.

A.2. Basic Assumptions about a Possibly Suggested X3D Component "SMUOS"

1. SMUOS will follow the enhanced model/module/frame paradigm (eMMF paradigm)

2. Basic assumptions about SMS models (*M*MF)
2.1. SMS models are represented using declarative 3D principles
2.2. Models can be rendered (they are visible, audible, sensible, ...)
2.3. Models are rendered relative to a module
2.4. If models are not attached to a module, then they cannot be rendered
2.5. SMS models contain MIDAS objects to become multiuser capable
2.6. MIDAS objects depend on the SMUOS component. They are specialized to distinct use cases (binary switch, carousel drive, ...)
2.7. MIDAS objects cannot be rendered. They make only sense together with a model or module

3. Basic assumptions about SMS modules (M*M*F)
3.1. An SMS module is NOT a tile
3.2. SMS modules are represented using declarative 3D principles
3.3. A module renders a (small) part of a virtual universe
3.4. A module spans a (pseudo) euclidean space time, where models and/or avatars can meet
3.5. The relation of a module's coordinate system to the world coordinate system of the Web3D browser is "for further study" (ffs.)
3.6. Modules can contain "intrinsic" models. "Intrinsic" models are implemented directly as a part of a module
3.7. Modules can contain "bound" models. "Bound" models are implemented separately from modules, so that they can be used by many modules.
3.8. Modules can contain "unbound" models. "Unbound" models can be loaded or unloaded independent of all modules. If loaded, an unbound model can be "assigned" to a module. Changing the assignment from one module to another is called "handover"
3.9. Each SMS module contains an instance of the "SMS module coordinator". The SMS module coordinator is a node defined by the SMUOS component. The SMS module coordinator cannot be rendered, but it coordinates all MIDAS objects of all models that are attached to its module

4. Basic assumptions about the SMS frame (MM*F*)
4.1. The frame integrates one or more SMS modules into a VR/AR platform
4.2. The frame is responsible to register, load or unload top level modules
4.3. SMS models cannot be contained directly in the frame. A module must be in between
4.4. If the scene shall be multiuser capable, then the frame
4.4.1. is responsible to initialize the used MU system and to provide a "sessionId" to the SMUOS component
4.4.2. must provide a network connection to the SMUOS component
4.5. The SMS frame contains an instance of the "Simple Scene Controller", which is a node defined by the SMUOS component.
The "Simple Scene Controller" cannot be rendered, but it controls the SMS module coordinators of all SMS modules and hence it controls the complete SMUOS component within a scene instance

5. Basic assumptions about "classic avatars"
5.1. Classic avatars have N instances, where one instance – the so-called "pilot" – is part of the scene instance that is used by the user, who is represented by the avatar, and the other N – 1 instances – the so-called "drones" – just follow the pilot
5.2. With the help of the special MIDAS object "Avatar Container", we can
5.2.1. assign avatars to the frame (world coordinate system)
5.2.2. assign avatars to some module (module coordinate system)
5.2.3. assign avatars to some model
5.3. Please be aware: SMS models can NOT be rendered relative to the world coordinate system. SMS models can NOT be rendered relative to SMS models
5.4. Hence classical avatars are NOT SMS models

6. Basic assumptions about "UM avatars" (unbound model avatars)
6.1. UM avatars are unbound SMS models
6.2. UM avatars are models that represent a virtual identity

7. Basic assumptions about moving modules
7.1. There will be a special MIDAS object, let's call it "Module Container", that will allow to instantiate SMS modules as parts of an SMS model
7.2. Hence the "basic MMF paradigm"
frame – 1:n – module – 1:n – model
will become the "enhanced MMF paradigm"
frame - 1:n – module – 1:n – model -1:n – module – 1:n – model ad. inf.