Zelestra
Core Framework Concepts

August 2004
Revised January 2011

Zelestra
Software Architecture for the New Media™

Table of Contents

1.0 Introduction

This document presents the foundational concepts of a core framework for the application of federated content collections as virtual worlds that are distributed across a loose federation of computing resources Virtual worlds may provide users a wide range of business, educational, or entertainment experiences, from small scripted activities, to large simulated environments. Virtual worlds bring together content, dynamic modeling, and user specific customization to provide an immersive interactive experience tailored to the individual. This document does not attempt to describe a specific virtual world, but to define the core terms and concepts behind the Zelestra Core Framework.

Standard document formats and protocols allow virtual worlds to be distributed across diverse computing resources yet deliver a consistent user experience. Each computing resource may be individually owned and operated. Distributed virtual worlds are constructed from four computing resource categories: digital libraries, user communities, world engines, and user clients.

Digital libraries provide multimedia resources which are the basis of virtual world content. User communities provide user identity, accounting, and customization of the virtual world experience. World engines join multimedia resources, dynamic modeling, and user customization, to bring interactive environments to user clients while tracking intellectual property rights and resource usage. User clients provide an interface to the virtual world through user interaction with a web browser, browser plug-in, and/or custom applications.

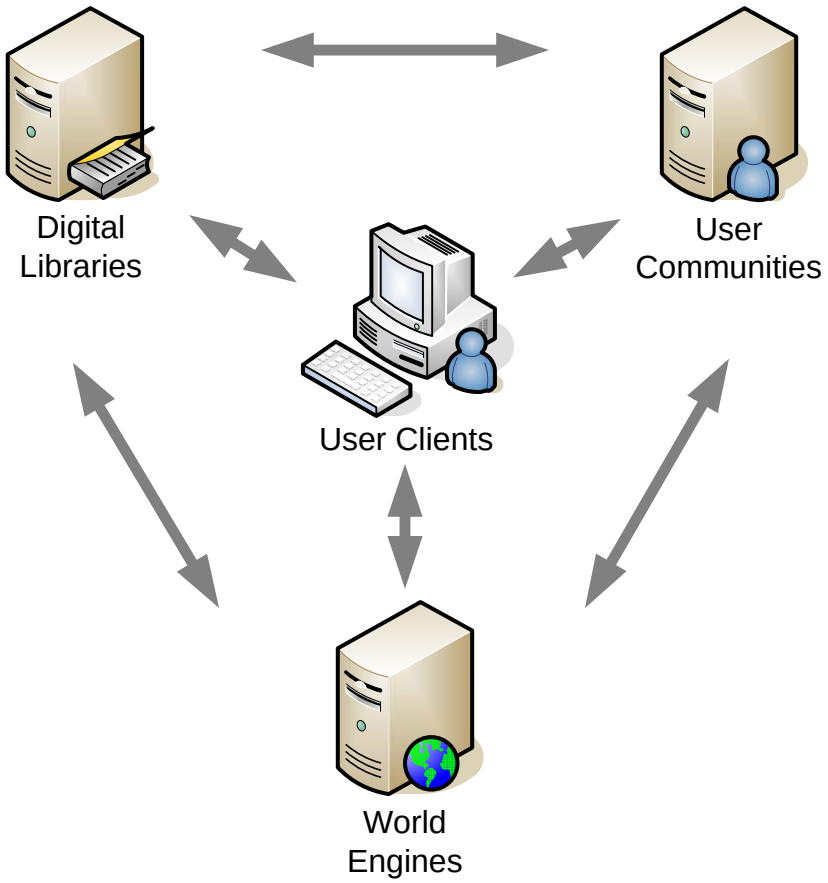

Figure 1.0

2.0 Digital Library

Digital libraries provide multimedia resources which are the basis of virtual world content. A digital library is an integrated set of services for capturing, cataloging, storing, searching, protecting, and retrieving information. A virtual world may have many digital libraries, each owned and operated independently.

A digital library accommodates heterogeneous participants, content, and users through adherence to widely adopted inter-operability standards. However, digital libraries are often stand-alone entities providing complete content services to their members and affiliations without regard to the repurposing of their content for virtual worlds. To be a content source within a virtual world, a digital library need only make the content available through a persistent and publicly available Uniform Resource Identifier (URI) using Hypertext Transport Protocol (HTTP) or Hypertext Transport Protocol over Secure Socket Layer (HTTPS). If content access requires user authentication, the content must be protected by a passive redirect-based Single Sign-On (SSO) protocol.

A mature digital library includes the ability to import data from content producers and export data to business and educational partners. Access to data and metadata is tracked and managed for each requestor and may be reported to the appropriate user community through web services. Metadata searches may be performed using both web pages and web services. Protocols such as Representational State Transfer (REST), Simple Object Access Protocol (SOAP), Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH), and Simple Digital Library Interoperability Protocol (SDLIP) are some examples of the standards that are utilized for these purposes.

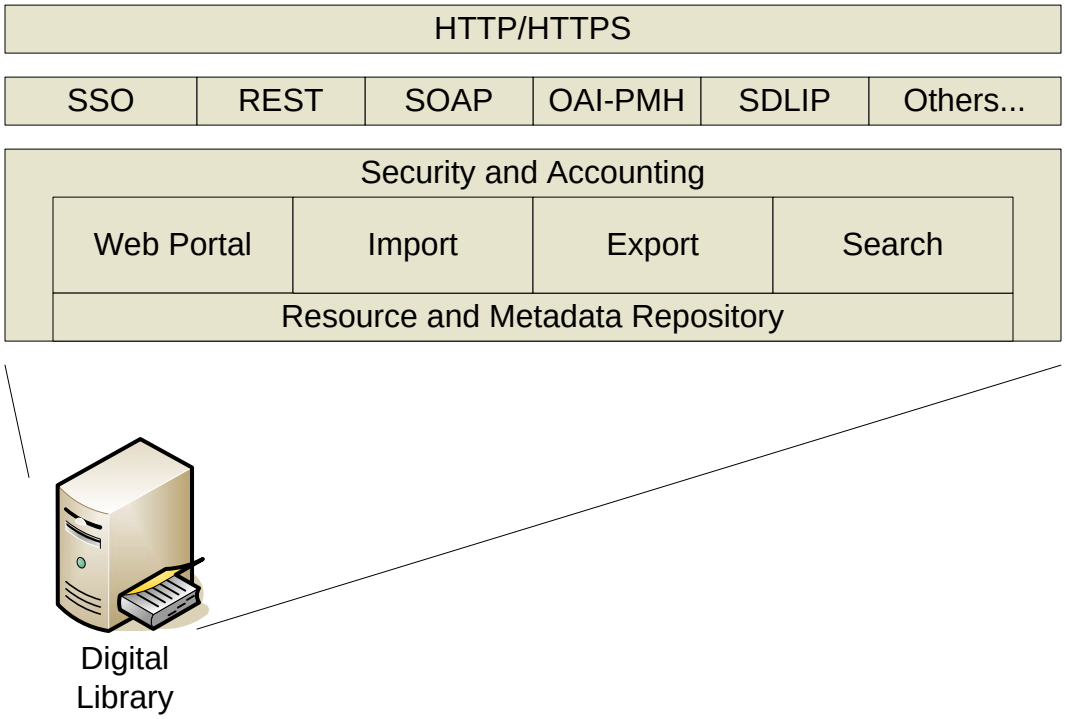

Figure 2.0

3.0 User Community

User communities provide user identity, accounting, and customization of the virtual world experience. A user community may also provide Web 2.0 collaborative environments for its members that share a common interest. For educational or research oriented virtual worlds, the user community is a place where team collaboration and aggregation of research takes place. In game worlds, the community may provide the game's back-story and a place to exchange tips and stories. User communities usually have many collaborative components, such as workspaces, chat rooms, mailing lists, etc. A virtual world may have many user communities, each owned and operated independently.

User identifiers and user preferences are managed by user communities and made available to digital libraries, world engines, and user clients as needed through adherence to widely adopted inter-operability standards. However, user communities are often stand-alone entities providing user authentication and complete collaboration services to their members and affiliations without regard to the repurposing of their platform for virtual worlds. To be a user community within a virtual world, a user community need only make user definition files available through persistent and publicly available URIs using HTTP, or HTTPS. The URI of the user resolves to the user's definition file and is the user's identity within the virtual world. If content access requires user authentication, the content must be protected by a passive redirect-based SSO protocol.

A mature user community may include web services to provide a means to dynamically update the user state and to aggregate access data from digital libraries and world engines. Protocols such as REST and SOAP are some examples of the standards that are utilized for these purposes.

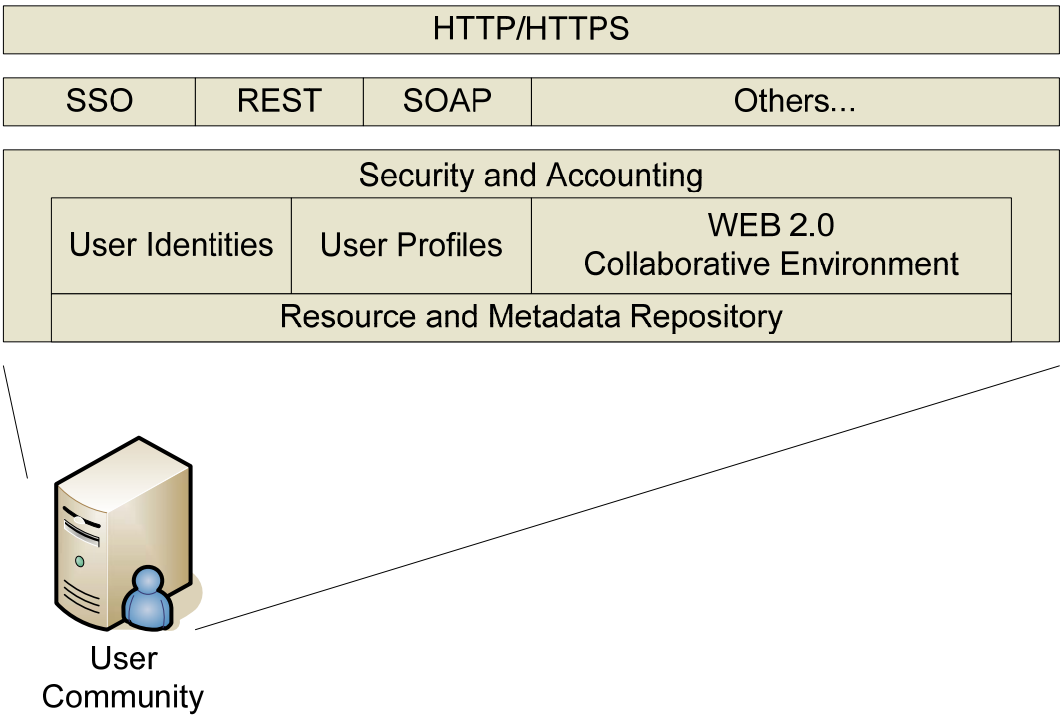

Figure 3.0

4.0 World Engine

World engines join multimedia resources, dynamic modeling, and user preferences, to bring interactive environments to user clients while tracking intellectual property rights and resource usage. A world engine holds the current state of all or part of the virtual world. A virtual world may have many world engines, each owned and operated independently.

A world engine manages specific virtual world object instances and their dynamic properties. This can be as simple as quiz results or as complex as an entire simulated environment. The engine exposes a web services interface to object methods to provide interactive state to a browser or user client application. Protocols such as REST and SOAP are some examples of the standards that are utilized for these purposes. Content and services are protected by a passive redirect-based SSO protocol.

Complex simulations that require persistent connections to user client applications are supported through socket based protocols such as Zelestra's Simulation Telemetry Transport Protocol (STTP) or Microsoft's DirectPlay.

By communicating with user communities, world engines can tailor instances to individual user's preferences and submit royalties and fees for payment.

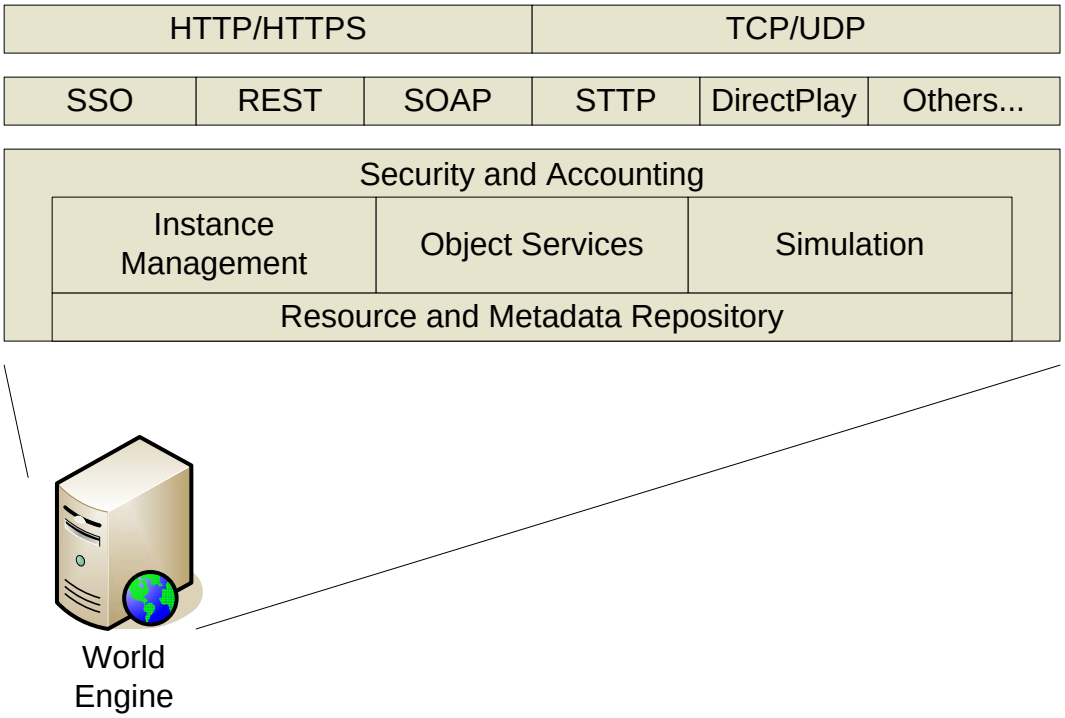

Figure 4.0

5.0 User Client

User clients provide an interface to the virtual world through user interaction with web browser and/or custom applications. User clients provide the hands-on interface between the user and the virtual world. User clients need not be personal computers; they may be cell phones, personal digital assistants (PDAs), or any other device that has internet connectivity.

A user client could be anything from a simple web browser, to a browser plug-in such as Adobe® Flash®, to a sophisticated 3D graphics application. Event-driven user clients that require persistent connections to world engines are supported through socket based protocols such as STTP or DirectPlay.

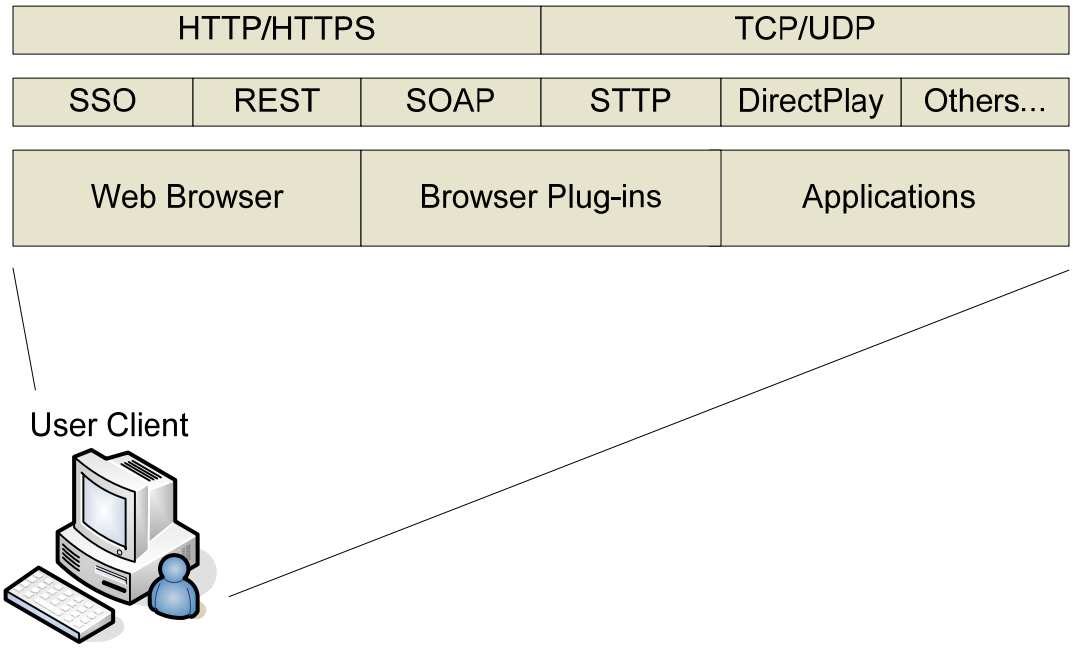

Figure 5.0

6.0 Real Space, Meta Space, and Virtual Space

Because describing objects and actions within virtual world design can get confusing, a good understanding of three related spaces is required. These three spaces are: real space, meta space, and virtual space. Real space is the real physical world. Meta space is the software that defines and supports the virtual space. Virtual space is the virtual world itself. Virtual space exists solely as a byproduct of the real space computer processing that takes place on meta space objects.

The confusion that can exist when discussing these spaces is especially apparent when describing user interfaces. For example, an individual may be the user named "John Doe" in real space, the user object identified as "jdoe" in meta space, the agent object identified as "agent53" in meta space, and the agent named "Explorer Jon" in virtual space. In addition, a user may have multiple user objects in meta space, each of which may support more than one agent in virtual space. It is therefore vital that the proper terms and spaces be recognized and used when discussing virtual worlds.

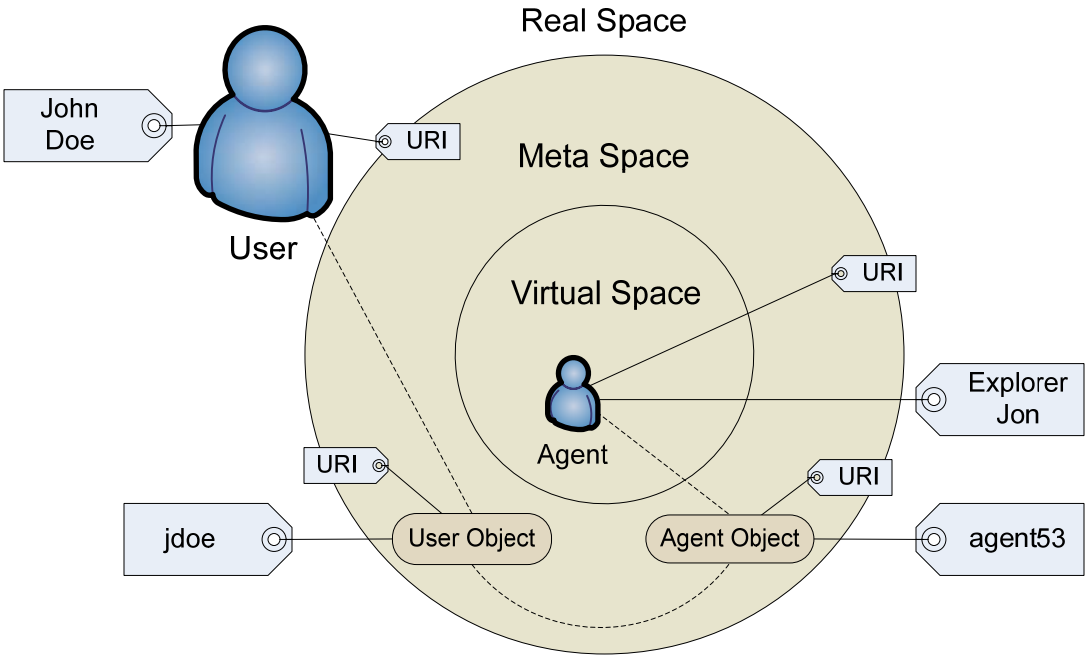

Figure 6.0

Generally, things in virtual space are called by their real space equivalents: tree, car, building, etc. The user can talk about how they "found an unusual species on an island" as if they were really there. In our example above, the user John Doe could say "I am an explorer and scientist". However, when talking about real or virtual space things modeled as meta space objects, the term "object" is added. So the user John Doe in real space is modeled in meta space by a user object, and the agent Explorer Jon in virtual space is modeled in meta space by an agent object. John Doe, jdoe, agent53, and Explorer Jon may each be identified by a unique URI.

In this document, the term "user" is used to describe a real space individual seated at a user client computer. The term "user object" is used to describe a meta space object that represents the user. The term "agent" is used to describe a manifestation (virtual presence) of the user in virtual space. The term "agent object" is used to describe a meta space object that represents the agent.

7.0 Datastream

A datastream is a stream of zero or more bytes in meta space identified by a URI and retrieved using HTTP or HTTPS. A datastream is of a particular Multipurpose Internet Mail Extensions (MIME) type. This includes such types as Hypertext Markup Language (HTML), Extensible Markup Language (XML), Resource Description Framework (RDF), image, sound, video, and others. Datastreams hold both the structure of a virtual world as well as its fundamental content. Datastreams are managed by service providers. Datastreams are retrieved by user clients and other service providers as needed. A service provider is identified by a domain name. The domain name of a service provider is also referred to as the provider identifier. Digital libraries, user communities, and world engines, are implemented on one or more service providers. If a datastream is to be secured, it MUST be protected by a passive redirect-based SSO protocol.

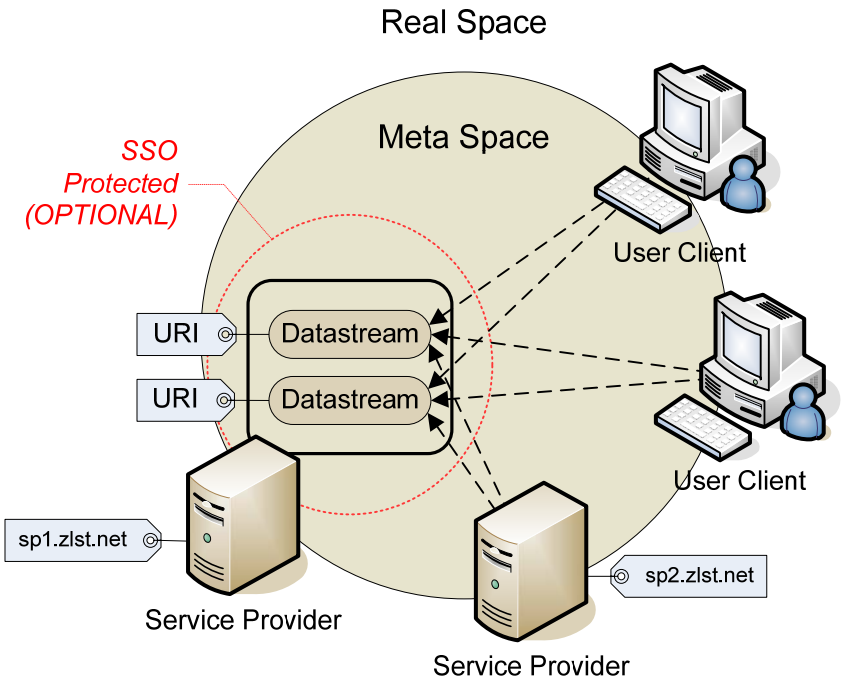

Figure 7.0

8.0 Resource

In its broadest meaning a resource is a conceptual entity which is identified by a URI. A meta space resource is one or more datastreams within meta space. If a resource is not a meta space resource, then it is a real space resource or a virtual space resource. All real and virtual space resources SHOULD be resolvable to a meta space resource that is a datastream representation of the real or virtual space resource. Real or virtual space URI resolution is accomplished using the hash or slash mechanisms recommended for Semantic Web development.

When the term resource is used without indication to which space the resource belongs, the reader must determine to which space the term refers by the context of the discussion. Most of the time within this document the term resource refers to a meta space resource, sometimes called an information resource.

When a resource consists of multiple datastreams, the MIME type requested determines which datastream is retrieved. This is usually accomplished through the HTTP request Accept header.

Service providers that store resources are called resource repositories. A digital library is a kind of resource repository that provides additional services such as the ability to perform searches. A virtual space may utilize multiple resource repositories. If a resource is to be secured, it MAY be protected by a passive redirect-based SSO protocol.

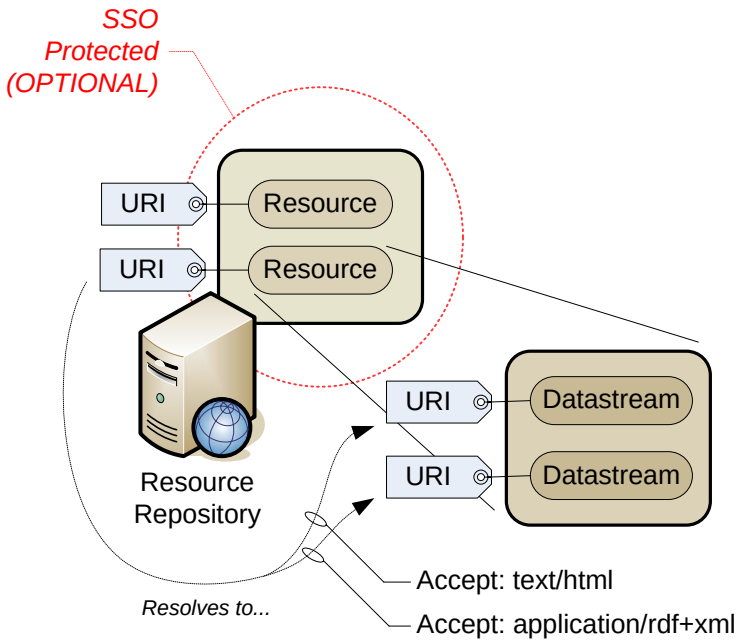

Figure 8.0

9.0 Virtual Space Identifier

Though a virtual world may be distributed across multiple service providers the user is presented with an integrated and seamless view of virtual space. The distributed nature of the supporting meta space is transparent to the user.

Multiple independent virtual spaces may be implemented across the same distributed infrastructure. These independent virtual spaces may be topographically identical, yet they do not share state. It is as if they are the same worlds in different dimensions of existence. Actions that occur in one virtual space do not affect any other virtual space.

Virtual space independence requires that all relevant computing resources on the network know the specific virtual space they are supporting at any given time. Because distributed virtual worlds are capable of being deployed across the Internet, an Internet-wide method of uniquely identifying a virtual space is required. This is accomplished by assigning a unique URI to each virtual space. This URI is also referred to as the virtual space identifier. A virtual space identifier is resolved to the definition document of the virtual space.

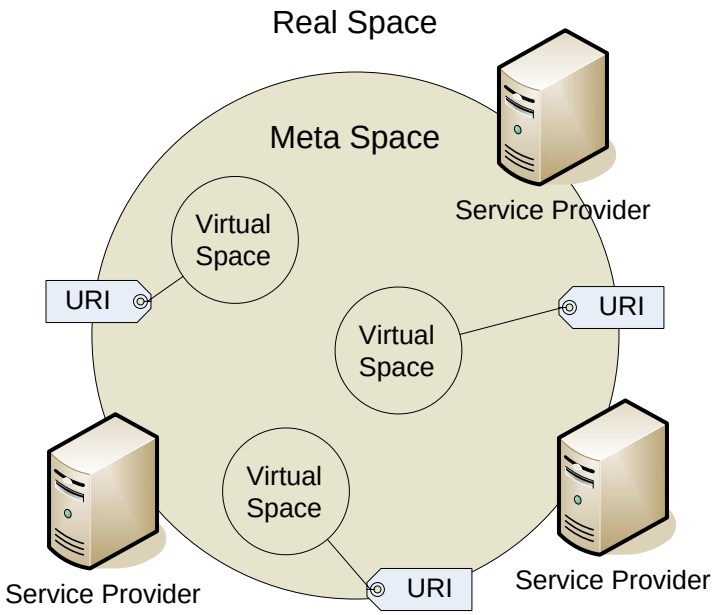

Figure 9.0

10.0 Instance

An instance is a kind of resource that is identified as being in the context of a particular virtual space. While the datastreams of a resource are shared across all virtual spaces, the datastreams of an instance are specific to a particular virtual space. An instance is identified globally to all virtual spaces by a single URI. However, the URI may resolve to a different set of datastreams depending upon the virtual space that is in context. In addition, an instance may usually be saved within a repository as a normal resource to provide the default contents of any future instances of that resource.

Instances are stored in resource repositories. Not all resource repositories support instances. Because a virtual space MUST be identified to support an instance, it is REQUIRED that an instance be protected by a passive redirect-based SSO protocol.

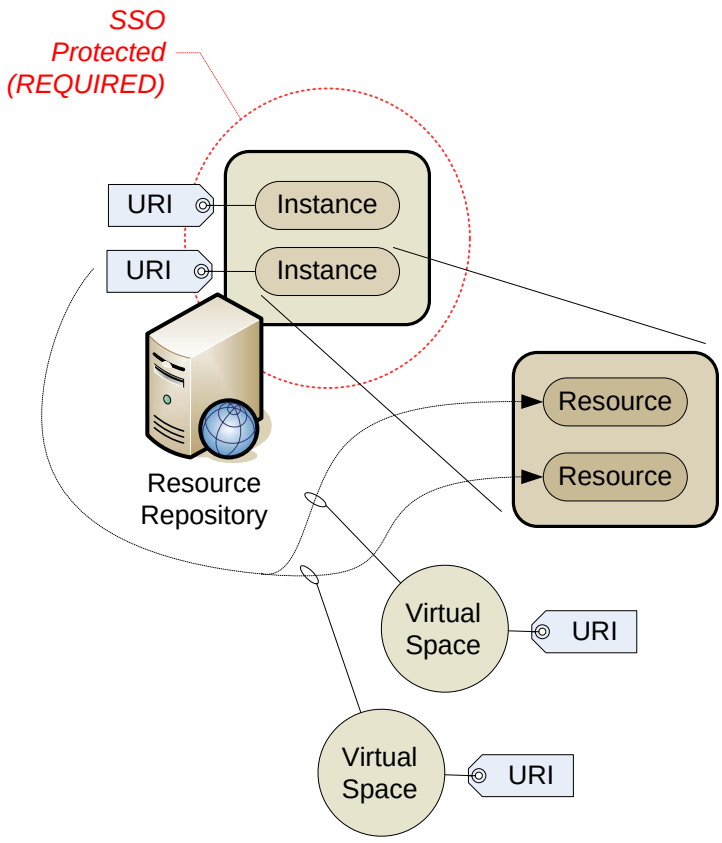

Figure 10.0

11.0 Realm Identifier

Though a virtual space is considered to be one integrated and seamless whole, the supporting meta space may be distributed across multiple service providers. Datastreams, resources, instances, and services usually fall into distinct sets that are implemented and secured as a group. These groups are called realms. A virtual space may therefore consist of several realms. A realm may in turn consist of several service providers. Communications with a realm is established through the web services available at one or more of the realm's service providers. Protocols such as REST and SOAP are some examples of the standards that are utilized for these purposes. When a service provider is REST-based, it MAY be protected by a passive redirect-based SSO protocol. Socket based-protocols first obtain their connection parameters through a realm's web services.

A unique URI is assigned to each realm. This URI is also referred to as the realm identifier. A realm identifier is primarily used to identify the realm to which an agent is currently attached. A realm identifier is resolved to the definition document of the realm.

Each virtual world has a realm that acts as a starting point for entry into the virtual world. This realm is referred to as the entry realm for the virtual world. The entry realm manages the initial selection of an agent for the user and to what virtual space the agent is assigned.

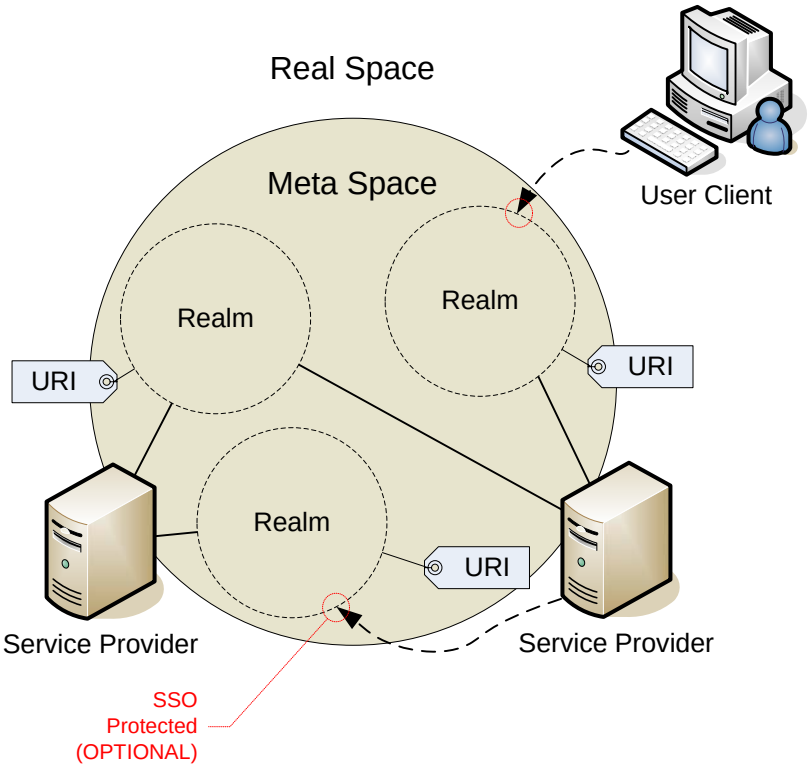

Figure 11.0

12.0 Habitat

A habitat is a location in virtual space that can contain an agent. In the example given in section 6.0, if the agent “Explorer Jon” is on a ship, then the ship is considered a habitat. Habitats can reference other habitats in prepositional relationships; the ship is on the ocean, which is on the planet, which is in the solar system, which is in the virtual space. Habitats need not be modeled after real space equivalents. They may be logical, such as “activity B, in biology unit three, for grade 5”.

When an agent is contained by a habitat, the corresponding agent object maintains a realm identifier of the realm that contains the habitat. This allows a user client to locate the realm that currently manages the state of the agent.

An agent may also be a habitat. For instance, if a user may control a ship directly, then the ship is an agent of the user. However, the ship is also a habitat in that it may also contain an agent that is the virtual human presence of a user. In this case, the relationship between the ship as a virtual presence defined by an agent (exterior) and ship as a habitat (interior) is maintained by the realm that manages the ship object.

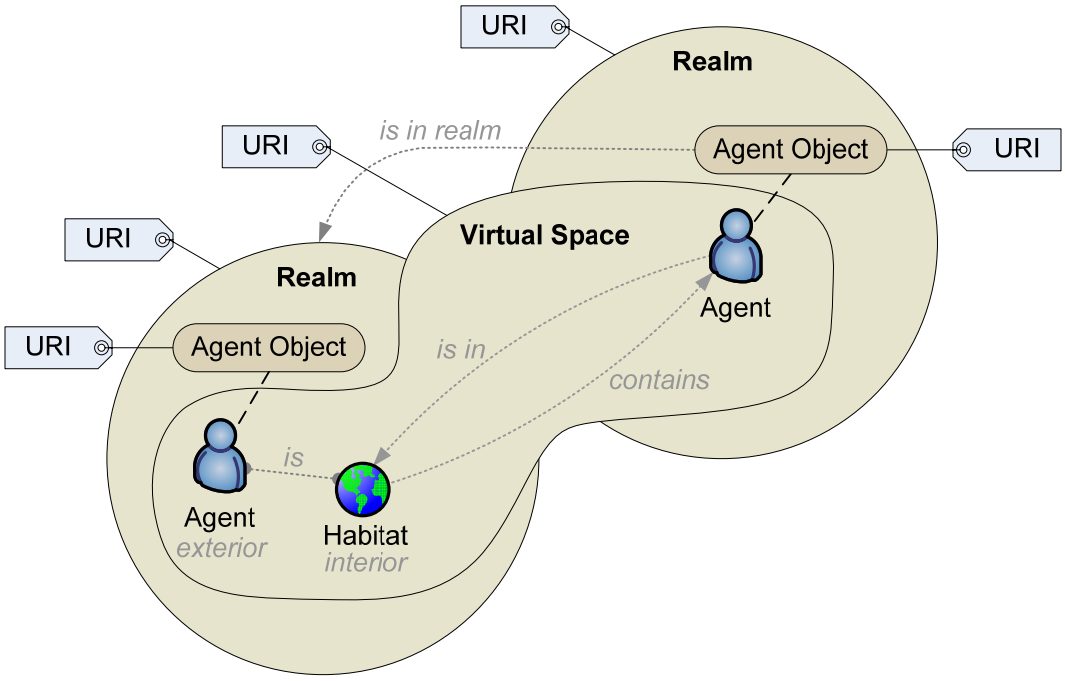

Figure 12.0

13.0 Definition Document

A definition document is a resource that defines a real or virtual object in meta space. Definition documents are documents that include a set of standard elements. Security access controls may cause a resource repository to withhold some elements when these documents are retrieved by certain resource requestors. Only optional elements may be withheld for security purposes. The standard elements are defined as follows:

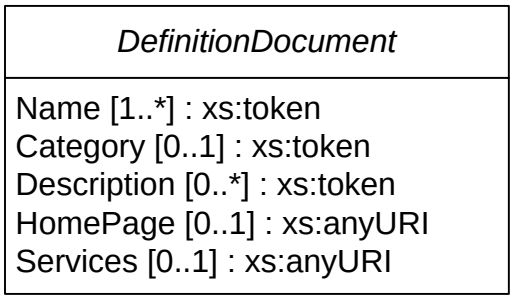

Figure 13.0

Name
This REQUIRED element is the name of the object within real or virtual space. This is often different from the title element supplied in the metadata of the object. Titles in metadata are often written as if they are aware of the existence of meta space. Names in definition documents are treated as normal proper nouns. Using the previous example, the metadata title for an agent object could be “John Doe’s Agent” where the name would be “Explorer Jon”. Language localization is supported.

Category
This OPTIONAL element is the sub-type of the object. The type itself is defined by the name of the root element of the definition document.

Description
This OPTIONAL element is a brief description of the object within real or virtual space. This is often different from the description element supplied in the metadata of the object. Descriptions in metadata are often written as if they are aware of the existence of meta space and may include technical details. Descriptions in definition documents are treated as the casual description of a real object. Language localization is supported.

HomePage
This OPTIONAL element is an HTTP URI to a web page for the object.

Services
This OPTIONAL element is an URI to a services definition document that defines SOAP, REST, or other web service interfaces to the object. If a REST service is secured, it MUST be protected by a passive redirect-based SSO protocol.

14.0 Principal Definition

A principal defines a real or meta space identity in meta space. Both user objects and realms are examples of principals.

A principal definition is an abstract definition that may include certain standard elements and methods as follows:

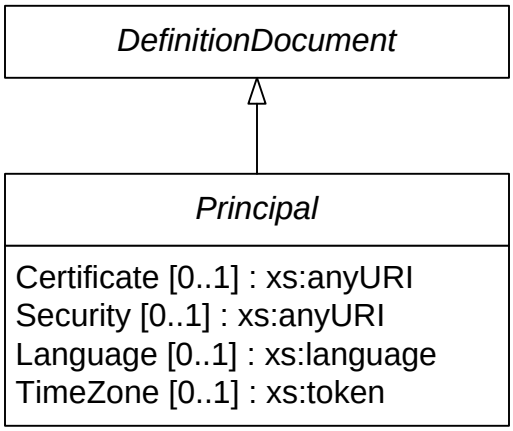

Figure 14.0

Certificate
This OPTIONAL element is a URI to a public copy of an X.509 digital certificate issued by a trusted authority. The certificate may be used to verify any signed messages that are generated by the principal. The data object pointed to by the URI MUST have a file name extension of ".cer". As per RFC 2585, the file MUST contain exactly one certificate, encoded in Privacy-Enhanced Mail (PEM) format with a MIME type of "application/pkix-cert". The URI of the principal MUST be included in the certificate as the subject common name.

Security
This OPTIONAL element is an URI to a security definition document that provides identity and/or service provider information for this principal.

Language
This OPTIONAL element is the preferred language of the principal.

TimeZone
This OPTIONAL element is the preferred time zone of the principal.

15.0 Agent Definition

An agent defines the virtual presence of a principal in virtual space. A principal may have many agents.

An agent definition defines the interface between a user client application and a virtual space. User client applications first obtain the virtual space identifier of an agent from the agent itself. Therefore, an agent MUST be a resource (not an instance).

An agent definition is a document that may include certain standard elements and methods as follows:

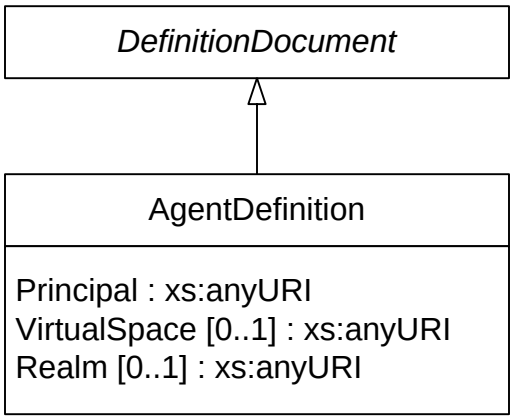

Figure 15.0

Principal
This REQUIRED element is a URI to the principal for this agent. Agents that are not the virtual presence of a user are usually the virtual presence of a realm.

VirtualSpace
This OPTIONAL element is the virtual space identifier of the virtual space which currently contains the presence represented by the agent. If the element does not exist, then the agent is currently not contained by any virtual space.

Realm
This OPTIONAL element is the realm identifier of the realm which currently contains the virtual presence represented by the agent. If the element does not exist, then the virtual presence is currently not contained by any realm.

16.0 Core Schemas Summary

The core framework depends upon the following core schemas:

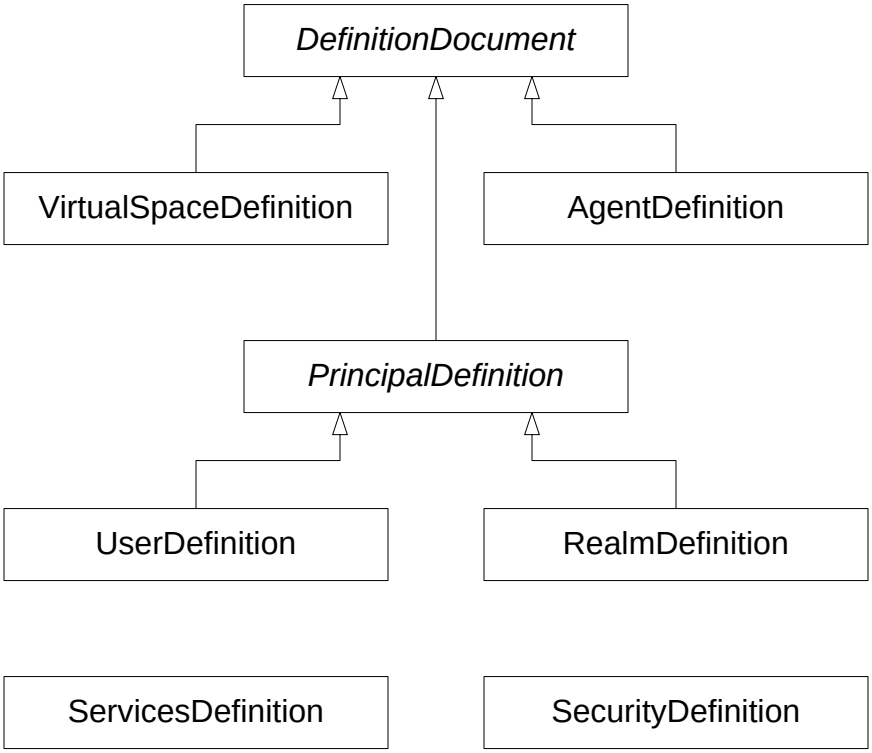

Figure 16.0

17.0 Core Associations Summary

The core framework has the following associations:

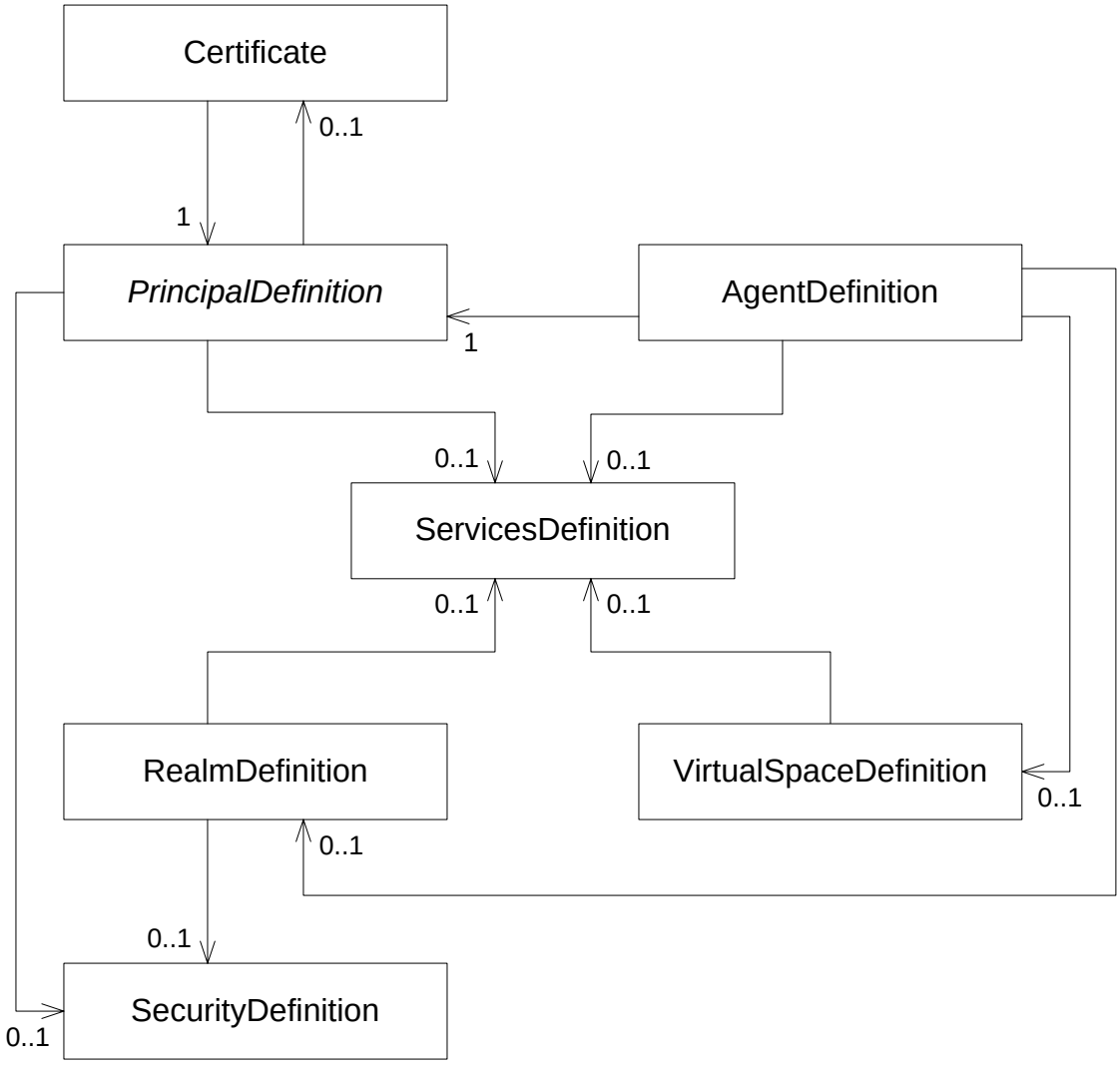

Figure 17.0

18.0 Profiles

There are three basic user client profiles that describe the use of the core framework for virtual world design. They are the web browser profile, the browser plug-in profile, and the enhanced client profile.

The web browser profile defines a virtual world that consists of web pages and their content. The browser plug-in profile defines a virtual world that is accessed through an in-browser content player such as Adobe® Flash®. The enhanced client profile defines a virtual world that is accessed through one or more applications on the user client. Although these three profiles are treated separately here, they may be combined in the typical virtual world design.

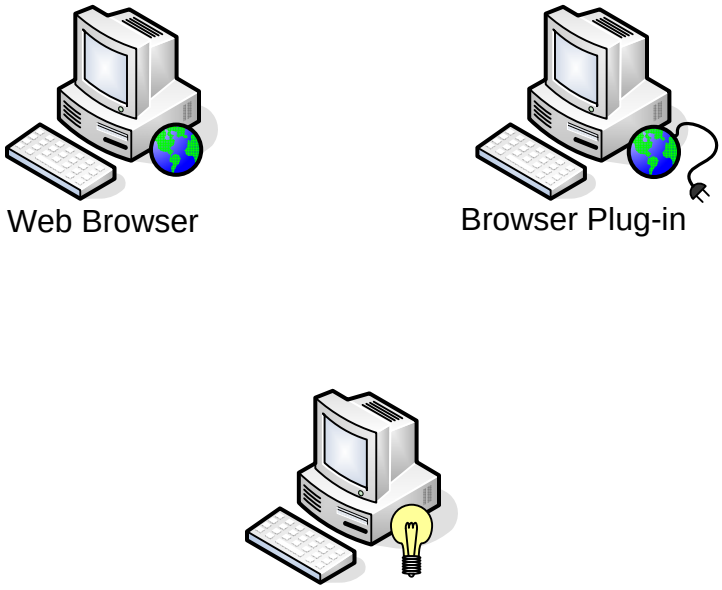

Figure 18.0

There are four general actions that users perform within the context of these three profiles. These four actions are:

- Joining a User Community
- Attaching to a Realm
- Interacting With a Realm
- Moving Between Realms

19.0 Joining a User Community

In order for a user to join a virtual world, the user must first become a member of a user community. The user community provides a federated user identifier that is acceptable to all participants of the virtual world. The user community may use passwords, digital certificates, third party authentication providers, or any other mechanism for establishing user client authentication. If the user community provides digital certificates, then the user community MUST provide the user a digital certificate with private key for download as a password protected Personal Information Exchange Syntax Standard (PKCS12) file.

The user community may direct the user to download and install the appropriate browser plug-ins or applications as required by the virtual world. When an enhanced client application is required, the user may also be required to download and install their PKCS12 file. Enhanced client applications may then use the URI imbedded in the subject common name of the digital certificate to obtain the user's identifier and then automatically download the user definition to complete the initialization of the application.

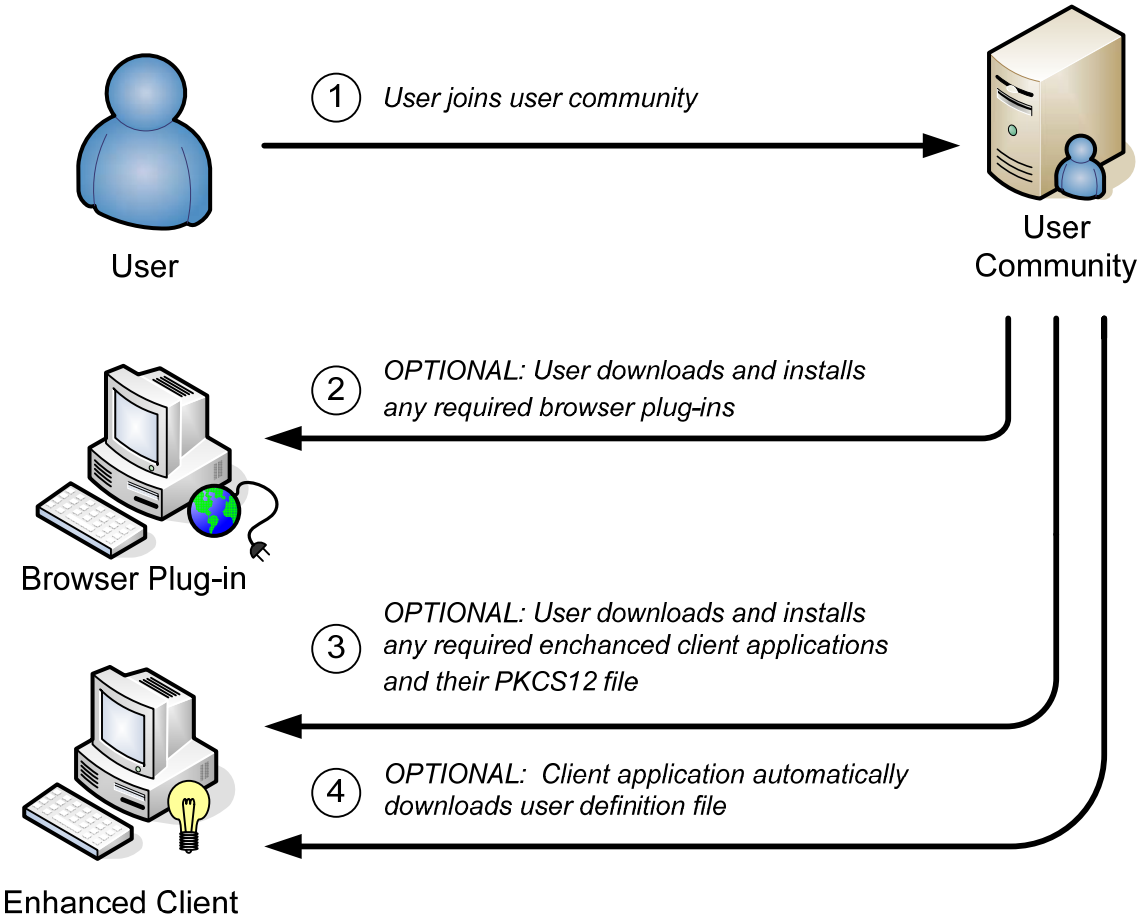

Figure 19.0

20.0 Attaching to a Realm

In order for a user to interact with a virtual world, user identity must be authenticated, a user client must be associated with an agent, and the associated agent must be attached to a realm.

The user client first contacts a virtual world at the virtual world's entry realm and logs into the entry realm. The user's user community provides SSO-based user identifier authentication. This is usually through a password or digital certificate. The service provider that manages user identifier authentication is referred to as the identity provider.

Once user identity has been authenticated, the user is supplied with an agent or optionally selects an agent to use. Under the web browser profile, agent selection would be accomplished through a browser session with the entry realm. The browser plug-in and enhanced clients may choose to download the user's agents from the entry realm through a web service.

Once an agent has been selected, the realm identifier within the agent definition is used to identify the appropriate realm. The services definition document referenced in the definition document of the identified realm is used to obtain the user client appropriate web services endpoint for the realm. Under the web browser profile, the user's browser would then be redirected by the entry realm to the appropriate realm's endpoint for browser clients. The browser plug-in and enhanced client may choose to download the agent definition, the agent referenced realm definition, and the realm referenced services definition document; then call the appropriate web services to communicate with the realm or to obtain the realm endpoint for STTP, DirectPlay, and other protocols.

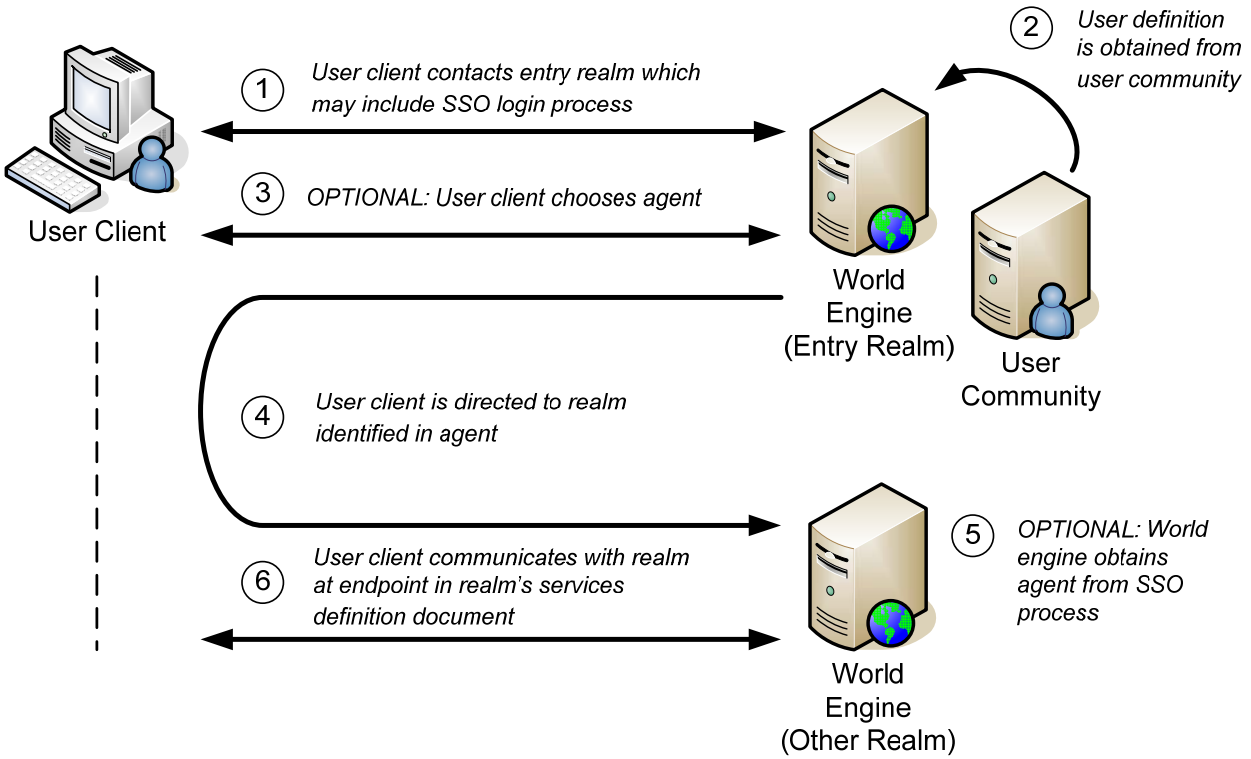

Figure 20.0

21.0 Interacting With a Realm

Once a user client is associated with a particular agent and that agent is attached to a realm, the user client is ready to interact with the realm and therefore with the virtual world itself. Virtual world state resides within various meta space objects, such as the user's agent, the current realm that the agent inhabits, and objects that reside in any world engines that may be affiliated with the current realm.

The realm's world engine directly manages virtual world state for objects maintained within the realm. Objects that exist within other realms are updated through web services defined by the services definition document referenced by their definition documents. Tracking of intellectual property rights, resource usage, royalties, and fees may be forwarded to user communities for aggregation.

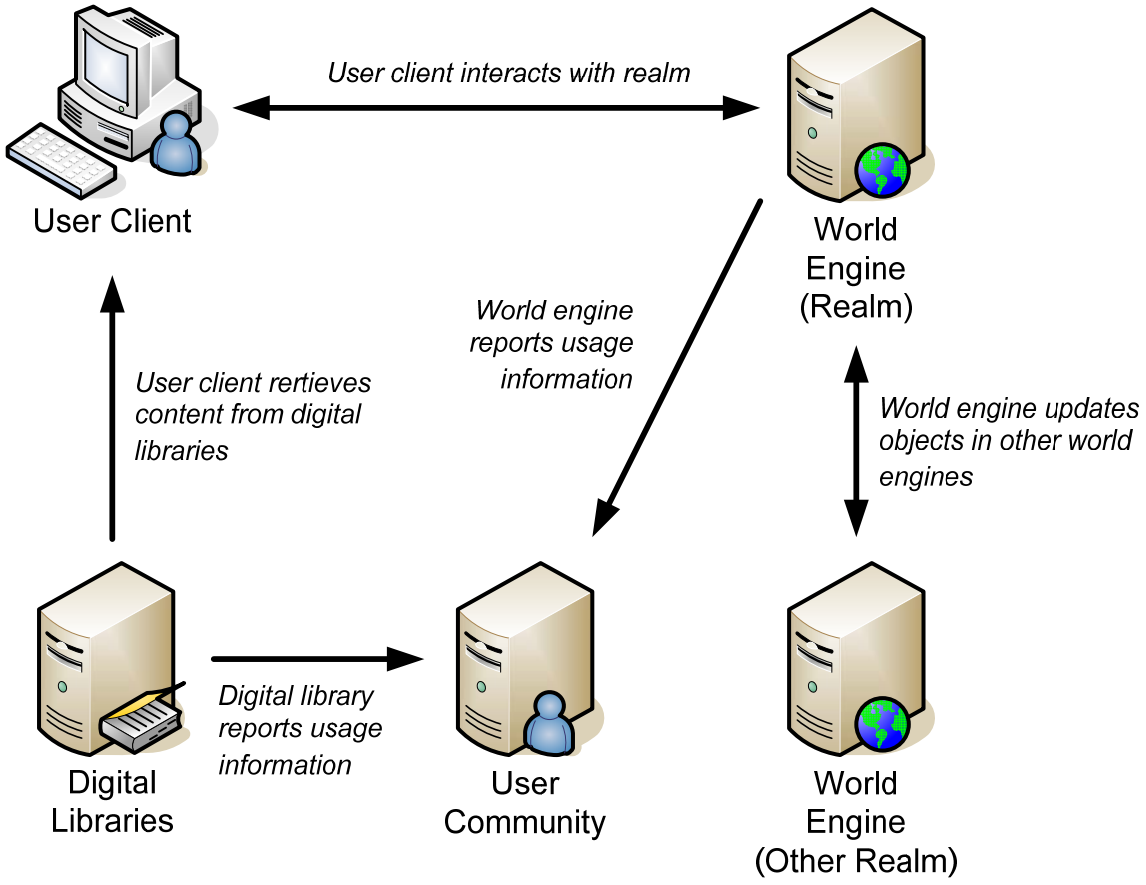

Figure 21.0

22.0 Moving Between Realms

Once an agent is attached to a realm, the realm has primary control over the state of the agent. Interactions between the user and the realm's world engine may cause primary control of the agent to transfer to another realm. This may happen by something as simple as the user directing their agent to walk through a door in virtual space or choose a different subject to study.

A virtual space object that has the ability to move the agent between habitats is called a portal. Moving between habitats within the same realm is managed internally by the realm's world engine. However, a portal object that moves the agent between realms must update the agent definition with the new realm URI and direct the world engine to orchestrate the transfer of primary control of the agent to the new realm.

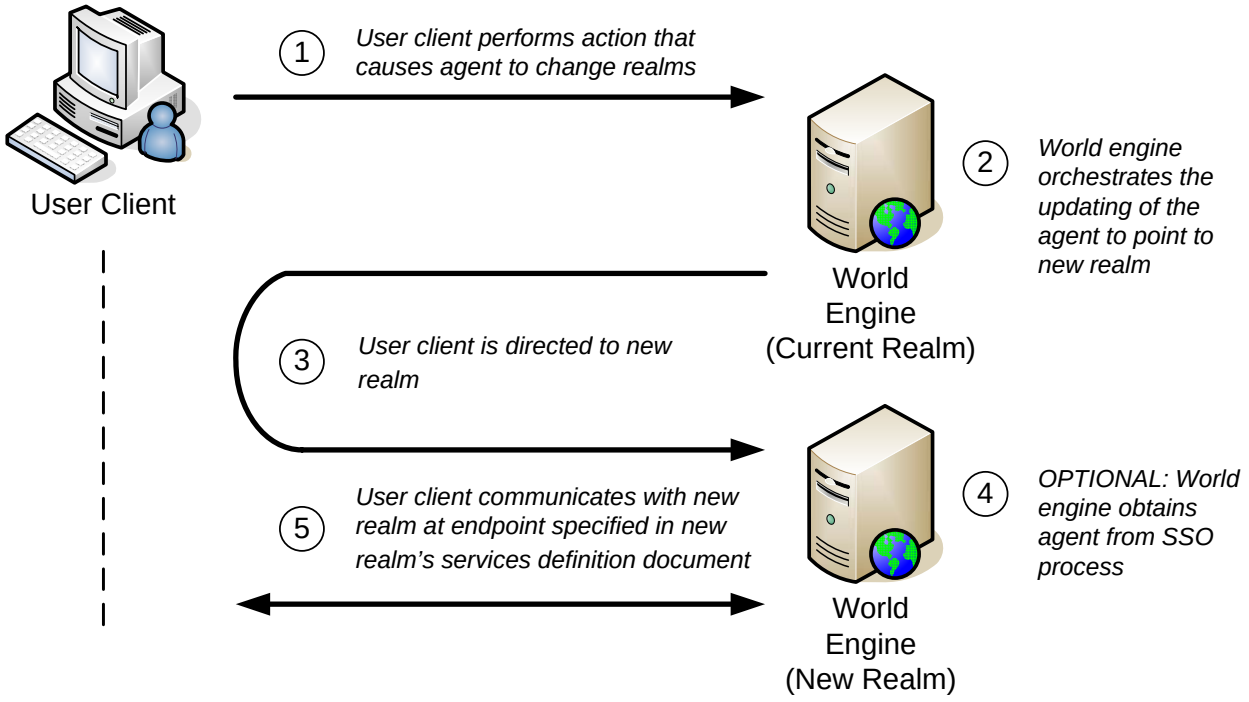

Figure 22.0

Index

