Internet Engineering Task Force	M. Hamrick	
Internet-Draft	Linden Research, Inc.	
Intended status: Informational	May 15, 2009	
Expires: November 16, 2009		

TOC

Open Grid Protocol: Introduction and Requirements
draft-hamrick-ogp-intro-00

Status of this Memo

Copyright Notice

Abstract

The Open Grid Protocol (OGP) defines interactions between hosts which collaborate to create an shared, internet scale virtual world experience. This document introduces the protocol, the objectives it attempts to achieve and requirements it imposes on systems and users

utilizing the protocol. This document also describes the model assumed by the protocol (to the extent it affects protocol interactions.)

Table of Contents

---

1. Introduction

TOC

Virtual Worlds are of increasing interest to the internet community. Innumerable examples of virtual world implementations exist; most using proprietary protocols. With roots in games and social interaction, Virtual Worlds are now finding use in business, education and information exchange. This document introduces the Open Grid Protocol (OGP) suite. This protocol is intended to carry information about the virtual world: its shape, its residents and manipulatable objects existing inside the world. The objective of the protocol is to define an extensible set of messages for carrying state and state change information between hosts participating in the simulation of the virtual world.

OGP assumes hosts operated by multiple organizations will collaborate to simulate the virtual world. It also assumes that services originally defined for other environments (like the world wide web) will enhance the experience of the virtual world. The virtual world is expected to be simulated using software from multiple sources. The definition of how these systems will interoperate is essential for delivering a robust collection of co-operating hosts and a compelling user experience. OGP describes interoperability expectations and mechanisms between systems simulating the virtual world and for service providers exposing their content to virtual world participants.

OGP presupposes a virtual world with the following characteristics:

The Virtual World exists independent of the participating clients.

This is in contrast to some systems which "call virtual worlds into being" as needed as a backdrop for social or task-oriented simulation. OGP assumes the state virtual world is normally "always on" and does not require a specific protocol to establish new virtual worlds.

Avatars have a single, unique presence in the virtual world.

The avatar, or the digital representation of an end user in the virtual world, has an existence that mirrors the common physical world; avatars (like people) do not exist in two places at once. Further, the avatar has a single, persistent identity that may be used to render a user-specific avatar shape or as the basis for access control.

The virtual world contains persistent objects.

Objects in the virtual world are governed by a "rational" life-cycle. They are created, persist and are (optionally) destroyed.

The OGP suite assumes that multiple hosts will participate in simulating the virtual world. Related to this assumption:

The virtual world may be partitioned.

> The virtual world is envisioned as being large; so large that it is impractical for a single system or cluster of systems to manage avatar presence, object persistence and physics simulation. The virtual world MAY therefore be partitioned to move services offered by different administrative domains onto distinct hosts. Virtual space may also be partitioned so that different "regions" of the virtual world are simulated by distinct hosts.

Presence, state and simulation happens on authoritative hosts.

> The presence, location and physical behavior of virtual objects and avatars are maintained and simulated by a host authoritative for a portion of the virtual world. This is in contrast to the "co-simulation" technique where each client maintains this information and communicates changes to each of its peers.

Version skew between simulation hosts be tolerable.

> The virtual world created by OGP is intended to be hosted on systems from several different administrative domains. It is unrealistic to assume that each administrative domain will run precisely the same version of the protocol. To protect against "brittleness" from version skew, the Open Grid Protocol uses a flexible object representation system known as LLSD. Used correctly, semantics of remote resource access may be maintained even when the participants in the protocol do not adhere to exactly the same revision of the protocol.

OGP uses Representational State Transfer (REST) style interaction over HTTP.

> Much of the protocol interaction between systems participating in the virtual world simulation uses a request / response interaction style. Rather than creating a new messaging framework, OGP layers much of it's protocol atop HTTP. Further, OGP uses Representational State Transfer (REST) like semantics when exposing a protocol interface.

A persistent, ubiquitous identity accompanies requests between hosts involved in the virtual world simulation.

> As in the consensus physical reality, each item is assumed to have a (largely) non-mutable identity. Unless acted upon by an external force, objects tend to retain their identifying characteristics (bricks remain bricks unless pulverized, etc.) Avatars too maintain an identity that allows the virtual world to properly render them.

1.1. Requirements Language

TOC

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 (Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," March 1997.) [RFC2119].

2. Open Grid Protocol Architecture

TOC

2.1. Protocol Objectives

TOC

The primary objective of the Open Grid Protocol is to provide a stable, extensible, secure system for virtual world information interchange with the following characteristics:

Identity is universal

Many network services are provided anonymously; the nature of the service does not require identity authentication prior to it's use. But with the increasing deployment of customizable services delivered on the internet, identity is increasingly important. Even services that contain information that might not be considered "sensitive" require a representation of digital identity if for no other reason than to match service requests with user preferences. For example, a web page presenting current weather information may be enhanced by remembering locations of interest to each user. Recent work with "web mash ups," where multiple personalized or sensitive resources are used in concert with one another points to the utility of a "universal" identity. The representation of this universal identity enables independent services to cooperate to present the facade of a unified application to the service consumer. This allows service aggregators to more easily integrate "best of breed" services into a consistent solution.

Universal identity is critical to the virtual world. To achieve an internet scale virtual world, user services must be distributed amongst multiple hosts. To achieve a compelling experience, it must be easy for service providers to deliver their services in the virtual world. To facilitate a compelling

social experience in the virtual world, all users must have the ability evaluate identity information of other users. Domains responsible for virtual world simulation MUST use a consistent representation of identity across all their hosts; simulation would otherwise be uncoordinated. Service providers who deliver content into the virtual world MUST use a consistent representation of identity to maintain the persistence of the virtual manifestation of their service; virtual objects used in conjunction with these services might otherwise appear to change state without apparent cause. Users depend on the persistent, universal identity of other users; if an avatar's identity changed unexpectedly, the result would be a suboptimal virtual world experience.

Flexible presentation of protocol data

While the primary purpose of the virtual world is to simulate a physical or social space, the tools used to access objects in the virtual world may be varied. Using a "3d viewer" is the primary mode of interaction with the virtual world, but other tools may be better suited for some tasks. For instance, it may be easier for a user to use a web browser to review avatar profile information, or to change details of virtual objects. Further, virtual world "mash ups" may prove to be important to some communities. To support the web (where XML and JSON are the lingua franca of information exchange) while also supporting tools where binary encodings are more appropriate, OGP was designed to be "presentation neutral."

OGP protocol exchanges are described in terms of an abstract type system using an interface description language. Implementations may choose to instantiate actual protocol data units using the most appropriate presentation format. Web-based applications may choose to use JSON or XML. Server-to-server interactions may use the OGP specific binary serialization scheme if implementers and deployers view binary encoding to be advantageous. The decision of which serialization scheme to use is ultimately that of the system implementer. OGP has been designed to provide this flexibility to system implementers and those tasked with deploying OGP compatible systems.

Flexible decomposition of concerns and ease of extension

OGP has been designed to allow meaningful separation of concerns. In other words, changes in one part of the protocol should not appreciably affect other parts.

For example, the authentication portion of the protocol is independent of the part of the protocol that deals with instant messaging or instantiating objects in the virtual world. In

addition to defining messages for communicating application state, the specification also defines pre- and post-conditions. Should one particular authentication scheme be found to be lacking, it can be modified or replaced without affecting other systems.

This type of separation of concerns in the protocol specification also makes it easy to deploy "related solutions." While OGP was designed primarily to communicate the state of the virtual world between servers and client applications, a number of related applications also exist. E-Commerce web sites related to the virtual world and mobile chat clients allowing instant messaging between mobile networks and virtual world participants are just two examples of such applications. Proper separation of concerns allows new services to be specified and deployed without the need to redefine existing protocol.

Resilience in the face of version skew

Core to the OGP protocol is the idea that different components and services may be operated by different administrative entities; identity management services might be operated one business while simulation services are operated by another. In environments where many different organizations participate, version skew can be an important concern. OGP was designed to "degrade gracefully" when two systems running different versions of the protocol attempt to communicate.

OGP uses the LLSD abstract type system to represent protocol data units (PDUs.) Because LLSD makes extensive use of variable width, clearly delineated data fields, consumers of PDUs may identify and extract only those PDU fields they know how to handle. While this is not a guarantee that message semantics may be preserved in all version skew situations, it does eliminate one important cause of interoperability failures.

2.2. Structural Architecture and the Role of Domains

TOC

The Open Grid Protocol assumes a division between systems offering user / avatar oriented services and systems offering virtual world simulation services. OGP was designed to support the case where the administrative authority for agent services is distinct from the authority providing simulation and object persistence services. The administrative authority of the former group is termed the "agent domain" while the latter is termed the "region domain." The protocol allows the agent domain and region domain to be distinct; in other

words, a user's identity may be managed by one person or organization while the virtual world they inhabit may be simulated by hosts owned by a completely different organization.

The motivation for this split is two-fold: First, it allows systems to scale along the two most independent axes (agent count and virtual world size.) Second, it moves identity management out of the domain of virtual world simulation, allowing the same avatar to be easily used in virtual world simulations managed by different administrative domains.

Each domain offers services to authenticated peers: user authentication, avatar and object presence, physics simulation, digital asset hosting, group messaging, etc. User authentication and avatar presence define the agent domain; they are it's raison d'être. Physics simulation and object presence define the region domain. Other services are assigned to the agent or region domain according to the expected scaling behavior, though their presence in a particular domain does not imply a hard and fast rule they may only exist in that domain. Digital assets, for instance, are expected to generally be under the administrative control of an agent domain. The digital asset service is thought to be an "agent domain service." However, some deployers may find it convenient to define assets belonging to a specific region as being a "region domain service."

It should also be noted that a client may consume services from multiple agent and region domains. The agent domain responsible for a user's profile and presence information may delegate responsibility for digital asset services, group messaging or user to user voice communication to a third party domain. It is expected that different parts of the same virtual world may be simulated on hosts from distinct region domains.

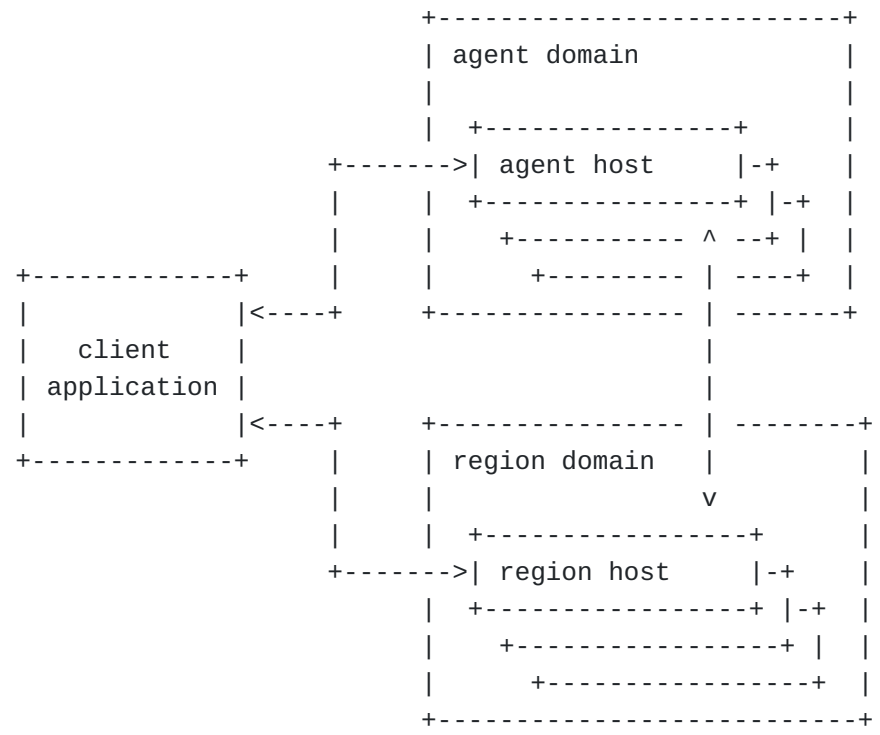

Figure 1: Protocol Flows in OGP

2.2.1. The Client Application

TOC

OGP presumes the virtual world is simulated for the benefit of human users. Whether that human is operating a "viewer" application to render the virtual world, or using a web interface to perform routine maintenance tasks, the user is expected to be operating software outside the administrative control of either the agent or region domain. OGP makes no assumptions about client software save it adheres to the described protocol.

2.2.2. The Agent Domain

TOC

The Agent Domain is the administrative entity that operates systems managing information about agents (i.e. - people) and related concepts. The agent domain is responsible for the following data and tasks:

*User and Avatar Profile and Identity Information

Virtual worlds are social spaces, used to interact with people. Information like the avatar's name, online friends and co-workers, group membership, personal and public notes and a user's list of "interesting places" in the virtual world are examples of the types of avatar profile information.

The agent domain may also wish to store information about the user: account name, contact information, billing information, etc.

*User Authentication

The first step in interacting with the virtual world is to authenticate the user's credentials (thus proving the user's right to control the avatar.)

Some system developers are interested in supporting completely anonymous avatars in the virtual world. Support for this feature is an option left to individual agent domains, but it should be mentioned that even if an agent domain wishes to support anonymity, the "authentication" step is required. In addition to demonstrating the user's right to control an avatar, the authentication step reserves server resources for use by the avatar. (see the discussion about seed capabilities later in this document.) Even if an agent domain wishes to support anonymous avatars, there is still a need to provide session specific shared secrets to ensure that anonymous avatars may not be "hijacked" by malicious virtual world participants.

*Avatar Appearance Information

Avatars in the virtual world are generally three dimensional figures represented using constructive geometry or polygon meshes. This information is of critical importance to client applications; without it, it cannot render the avatar.

*User Groups

Because virtual worlds are (often) social spaces, many systems support user groups to facilitate messaging and permissions amongst avatars who share similar interests.

*Individual or Group Text and Voice Chat

In keeping with the theme of virtual worlds being social spaces, agent domains may wish to support text, voice or even video chat between two users or amongst a group of users.

*Digital Assets at Rest

The virtual world is filled with virtual objects: conference tables, houses, airships, dragons, etc. These objects may or may not be instantiated in the virtual world. When objects are "at rest" or not being simulated in the virtual world, objects that are owned by a particular user are stored in an asset server associated with the agent domain.

*Avatar Presence

When a user authenticates themselves and wishes to interact with the virtual world, the user's avatar presence must be established. Simulating a user's avatar is a task shared by the agent domain, the region domain and the client application. The agent domain is responsible for establishing the user's presence in the virtual world simulated by the region domain, and to provide information about the avatar so it may be properly rendered.

2.2.3. The Region Domain

TOC

The Region Domain is the administrative entity that operates systems managing information about virtual land and related concepts. The region domain is responsible for the following data and tasks:

*Object Presence

The state of objects in the virtual world (landscapes, avatars, virtual "things") must be communicated to all participants. It is the responsibility of the region domain to keep track of each object's state. The landscape can change; clouds in the virtual sky may move and change shape; virtual people may move around and virtual "things" can undergo any number of state changes. The region domain is responsible for receiving input from virtual world inhabitants, evaluating how that input changes the state of objects in the virtual world, and then communicating those state changes to other observers.

*Physics Simulation

Simulating the physical behavior of objects in the virtual world is a core feature of any virtual world. Regions may choose "earth like" conditions, or may modify gravity and atmospheric settings to create the experience of being on a different planet. Still other options include simulating quantum effects seen at very small scales or the large scale relativistic effects seen on

galactic scales. Whatever the "physics" involved, it is the individual hosts in the region domain tasked with the simulation.

*Effects of Programmatic Changes

(aka "scripting.") Some virtual worlds allow users to modify the state of objects using simple programming languages. The region domain is generally responsible for managing and executing scripts that modify the state of objects.

*Region Specific Asset Storage

Though most "assets at rest" are associated with an avatar, it is conceptually appropriate for some items to be associated with a region; textures associated with landscapes, for instance. Or in some situations, the operator of a region may wish to bind "sensitive" resources to the location to ensure they do not follow region visitors to regions outside the originating region's administrative authority.

2.2.3.1. Protocol Flows

TOC

OGP defines protocol between the three architectural components above: the client application, the agent domain and the region domain.

> User authentication
> Before the agent or region domain expose service endpoints providing access to sensitive resources, the user operating the client application must be authenticated. The OGP Authentication (Chu, T., Hamrick, M., and M. Lentczner, “Open Grid Protocol: Authentication,” March 2009.) [I-D.hamrick-ogp-auth] specification describes the process of authentication between the client application and the agent domain.
>
> Digital Asset Access
> Responsibility for digital assets is shared between the agent and region domains. Digital assets "at rest" may be stored in an asset server associated with an agent domain. The agent domain exposes an interface allowing an asset's owner to manipulate some of that asset's metadata. The region host (or simulator) uses the same interface to retrieve the digital representation of the asset so it may be scheduled for simulation. Though the interface is the same, the asset server may trust the region host with sensitive data that may not be exposed to the client interface. After the asset is "rezzed" in world, the region host exposes an

interface client applications use to receive a description of the asset and updates to its state.

Avatar Placement and Movement
After a user is authenticated, their avatar may be placed into the virtual world (a process described as "rezzing".) After the avatar is "rezzed in-world", responsibility for its simulation may move from one region host to another. Initial placement and movement in the virtual world is an intricate interaction between hosts in the agent domain (which maintain information about the avatar's presence) and hosts in the region domain (which simulate the avatar and communicate its actions to client applications.) Initial placement is initiated by the client application, communicated to the agent domain which then communicates the request to the region domain on the client's behalf. Movement is usually initiated by the client and communicated to the region domain. If an avatar moves out of the virtual world region managed by a particular simulator and into a new simulator, the client must initiate the transit to the new simulator. The agent domain then contacts the new region, moves the avatar's presence there and removes it from the initial simulator.

Object Update
When an avatar initially enters a region, the agent domain provides it with an interface it may query on the region host to begin to construct the scene graph maintained by that simulator. Object state changes (movement, rotation, texture change, etc.) are communicated to the client application from the region host using a related interface.

2.3. Architectural Elements

TOC

OGP utilizes a number of "architectural motifs" or recurring design patterns. Most notably they include:

- exposing application state via RESTful resources
- using URIs to represent the address of application resources
- using HTTP to "carry" message oriented protocol data
- defining application state transitions and accesses with an interface description language (called LLIDL)
- using an abstract type system (called LLSD) to define access semantics of fields in protocol messages

*using multiple "serializations" of the abstract type system to support different categories of consumers; defined serializations include XML, JSON and Binary.

2.3.1. Communicating Application State Using REST-Like Resource Accesses

TOC

Contrary to popular opinion, not ALL virtual world interactions must be real-time exchanges. Many common activities like user authentication and texture and object transfer do not require "real time" semantics in the same way that applications like video-conferencing and Voice Over IP (VOIP) do. While it is generally a better experience if textures download quickly, if they are delayed, it does not have the same ramifications as if a voice packet in a VOIP system were delayed. Additionally, some interactions with the virtual world are strongly reminiscent of the request / response semantics used by popular protocols (like HTTP, POP3, etc.)

Because many protocol exchanges in the virtual world may be represented as non-real-time request / response interactions, OGP "reuses" the messaging semantics of HTTP. The justification for this is simple. Were OGP to not use HTTP, many of the features of HTTP would need to be re-invented or at least re-specified. Features like the use of mime types to identify payload structure; the use of message headers to modify the request or response and the use of URIs to address and identify resources. HTTP also has the benefit of being well supported by tools vendors and well understood by manufacturers of networking equipment.

Protocol exchanges in OGP that utilize request / response semantics are described using the LLSD / LLIDL abstract type system [I-D.hamrick-llsd] (Brashears, A., Hamrick, M., and M. Lentczner, “Linden Lab Structured Data,” February 2009.). LLSD defines type semantics for elements in a protocol data unit as well as rules for converting the data into a serialized form suitable for transmission across the network. OGP defines HTTP (and HTTPS) as the transports for serialized PDUs.

Addressable protocol endpoints in OGP are represented as URIs [RFC3986] (Berners-Lee, T., Fielding, R., and L. Masinter, “Uniform Resource Identifier (URI): Generic Syntax,” January 2005.). Protocol endpoints generally address RESTful resources. The OGP protocol uses HTTP verbs to provide read and write access to resources which represent the application state of the remote peer.

To recap, the objective of OGP is to communicate application state about the virtual world to all participants. OGP messages that communicate request / response style messages flow between clients and servers, using HTTP(S) as a message transport. Application objects representing the application state expose a RESTful interface and are

addressed unambiguously using URIs. The OGP PDU formats are described using LLIDL, the interface description language defined as part of the LLSD abstract type system. LLIDL defines RESTful resource accesses in terms of the LLSD abstract type system, which may be serialized using one of three well defined serialization mechanisms: XML, JSON and Binary. Protocol participants decide before interacting which serialization mechanism is most appropriate or use the content negotiation mechanisms defined in HTTP.

2.3.2. Bi-Directional Messaging with the OGP Event Queue

TOC

Not all protocol interactions are easily represented by HTTP's request / response semantics. When the server has a message for the client, there is no widely deployed technique for the server to initiate a HTTP request to the client. It is interesting to note that this is the same problem developers of "rich web applications" see when deploying their applications. Though OGP is not targeted for implementation exclusively in web browsers, we can utilize some of the techniques common in COMET applications.

Work is ongoing to define a general solution for "reverse HTTP," but many of these solutions require the definition of new protocol and deploying new code to web servers. The current best practice for COMET-style interaction is the use of the "long poll."

To avoid "technology lock in," OGP defines an Event Queue abstraction that represents the flow of messages from the server to the client. The Event Queue is expected to be implemented using the long poll technique. When additional options such as Reverse HTTP or web sockets are specified and in general deployment, the Event Queue may be re-implemented using these techniques. However, the interface defined by the Event Queue in the OGP Base document should not change. [I-D.lentczner-ogp-base] (Lentczner, M., “Open Grid Protocol: Foundation,” March 2009.)

OGP is intended to be sufficiently generic in its message definition that mechanisms for carrying OGP messages over other protocols might be defined. Protocols like XMPP [I-D.saintandre-rfc3920bis] (Saint-Andre, P., “Extensible Messaging and Presence Protocol (XMPP): Core,” March 2009.) offer good support for situations involving a message stream. Bindings for OGP over CMS over SMTP [RFC1822] (Lowe, J., “A Grant of Rights to Use a Specific IBM patent with Photuris,” August 1995.) are not outside the realm of possibility should the access of virtual world resources via electronic mail seem useful.

TOC

2.3.3. Using Capabilities to Simplify Inter-Domain Access Control

Simulated objects and services delivered by OGP compliant systems will require some level of access control. Unfortunately, distributed access control is a notoriously difficult problem. OGP seeks to minimize the drawbacks of distributed access control by use of capabilities. In this context, a capability is an opaque URL, some portion of which contains a securely generated, cryptographically unguessable sequence of digits. Capabilities are used to define service endpoints and are intended to only be in the possession of trusted parties.

For example, a system may export the capability:

http://www.example.org/s/B2A2A445-D234-463A-BE6D-6C54E5854FE4/

This URL defines the protocol endpoint used to communicate application state changes (or query application state) for a specific application object by a specific user (or delegate.)

Capabilities are required to be effectively unguessable as they represent the right to perform a set of operations on a particular resource. Additionally, they must be kept "secret." While the task of maintaining the confidentiality of a number of web resource addresses may be a burden, it does have the advantage of simplifying access delegation. If a subject wishes to delegate access to a third party, they simply communicate the capability.

To reduce the likelihood of successful guessing attacks, inadvertent disclosure of a capability and "time of check, time of use" attacks, capabilities in OGP have a fixed lifetime, after which they expire. Systems SHOULD pick capability lifetimes commensurate with their security requirements and MUST NOT respond to protocol requests directed at a capability's URL after it has expired. Additionally, OGP capabilities may be "single use" or "one shot," meaning that they may only be used once before expiring.

Because capabilities are randomly generated with a short lifetime, OGP defines a mechanism for securely communicating capabilities and re-requesting expired capabilities.

It is important to note that capabilities do not completely replace traditional access control models. Systems may still use traditional Subject-Permission-Object schemes to represent access control for objects under their control. Capabilities provide a mechanism for communicating access control decisions among loosely coupled trusted systems.

2.3.4. Using LLSD to Avoid Version Skew

TOC

It is a common practice in large, complicated software systems to divide the system into smaller, more manageable pieces. The precise nature of this partitioning is beyond the scope of this protocol. However, practical experience has demonstrated that services

distributed across multiple co-operating hosts MUST contend with the issue of version skew. Simply stated, version skew is the condition where multiple versions of a service are interoperating simultaneously. There are many reasons why version skew may be introduced. In OGP, agent domain hosts and region domain hosts may be operated by different organizations with different deployment schedules. Or perhaps a domain operator is required to support an obsolete version of a particular service endpoint for a small number of customers. Whatever the cause of version skew, it has, in the past introduced difficulties in deploying distributed services.

OGP does not seek to eliminate version skew, but it does attempt to reduce it's impact. OGP services are defined in using the LLIDL interface description language. LLIDL defines the type semantics of fields inside a protocol message using the LLSD abstract type system. Each of the abstract types defined in LLSD has a default value, and common conversions between conformable types are defined. LLSD specifies three standard techniques for serializing a protocol message prior to transmission across the network. Each of the three serialization techniques renders protocol messages into a collection of variable length fields. Protocol content is identified by JSON syntax, binary tags or XML element semantics, not by it's position in the message. LLIDL does not support the concept of a "required field." If a field defined in a protocol interaction is not present in the serialized message, it is semantically equivalent to the field being present and containing the default value for the field's type.

Careful construction of service endpoints allows them to consume messages described using LLIDL without fear that version skew induced format differences may cause the semantics of the message to be unclear. If a message arrives at a service endpoint with extra fields (fields defined in a later revision of the protocol exchange), the consumer can still extract those fields it understands. If a message arrives lacking a field described in the protocol exchange, the service endpoint SHOULD interpret it as if the field was present and contained the default value for it's type. This implies the message consumer cannot depend on the format of the message to determine validity, but must examine the contents of the message, converting missing fields to present fields with default values, and then determine if sufficient information is present to imply semantics about the protocol exchange. This technique will not eliminate all ramifications of version skew, but carefully constructed service descriptions should be able to avoid the most common problems found when services interoperate with minor revision differences. While the Open Grid Protocol itself does not mandate this style of message interpretation, it does require that messages be constructed so that service endpoints may do so.

TOC

3. Services Defined by the Open Grid Protocol

3.1. User Authentication

TOC

User Authentication in the Open Grid Protocol is intended to verify the user's authorization to control their avatar in the virtual world and associated services. OGP currently defines three methods for authenticating a user, as well as recommendations for integrating some third party authentication schemes. The inputs to authentication are an avatar or account identifier and a related authentication token. Assuming the token is successfully authenticated, the output of authentication is a seed capability or "seed cap."

Like most OGP protocol exchanges, authentication protocol data is represented as LLSD serialized data carried over a secure HTTPS transport. The use of TLS with OGP authentication is recommended for all deployers who do not employ some other network security scheme (IPSec, link encryption, etc.) Implementers are advised that in addition to user's password (or other credential,) the seed capability returned after successful authentication is also considered "sensitive" and should be protected with appropriate network security measures.

The three authentication schemes defined in the OGP Authentication specification [I-D.hamrick-ogp-auth] (Chu, T., Hamrick, M., and M. Lentczner, “Open Grid Protocol: Authentication,” March 2009.) use a cryptographic hashes to demonstrate the user is in possession of the shared secret associated with their account. Recommendations also exist for using transport authentication mechanisms (such as TLS client certificates) in place of shared secrets. Also, work is currently underway to define protocol messages for use with Secure Remote Password (SRP).

The authentication mechanisms described above are believed to be sufficient at the time of this writing. It is an unfortunate truth, however, that cryptographic primitives are occasionally shown to be less secure than originally believed. For this reason, OGP Authentication was designed to be extensible; allowing future users to define new authentication schemes without invalidating other authentication components. A further benefit of flexibility is the ability to integrate other authentication schemes into an OGP context. OpenID and SAML, for instance, are popular identity and user authentication technologies that are defined outside the IETF. OGP's flexible authentication system allows organizations responsible for these standards to define their use with OGP without having to change the text of the OGP Authentication standard.

A typical flow of events for user authentication follows. This is a simplified version; readers with an interest in authentication are referred to the OGP Authentication specification.

[I-D.hamrick-ogp-auth] (Chu, T., Hamrick, M., and M. Lentczner, “Open Grid Protocol: Authentication,” March 2009.)

1. The end user presents their account identifier (either avatar name or account name) and an authenticator to the authentication services of the agent domain. Endpoints for user authentication protocol messages are typically well defined, public URLs.

2. The authentication service authenticates the authenticator. If the credentials cannot be authenticated, an error condition is returned.

3. The authentication service generates a seed capability and returns it to the user.

4. The user queries the "seed cap," requesting capabilities for other services the user is authorized to use.

It is important to note that in the last step listed above, the client is free to request a subset of services offered by the agent domain. This allows the same authentication service to be used by restricted clients (for instance, a group-chat only client) as well as traditional 3d viewers.

3.2. Presence in the Virtual World

TOC

"Presence" in OGP refers to at least two related concepts: account presence and avatar presence. "Account Presence" describes the readiness for interaction between a user and an agent domain. A client applications signals the user's readiness for interaction with an agent domain's services by initiating (and completing) user authentication. Once authenticated, the user is "present." But an agent domain may export more services than interacting with the virtual world. It is conceivable a user may simply wish to manipulate their profile data, reorganize their digital assets, or make use of messaging services exported by the agent domain. Interacting with these services requires only "account presence." This type of presence implies only a client application presented legitimate credentials to the agent domain's authentication service.
When a user wishes to interact with the virtual world, their avatar must be placed or "rezzed" there. Placing an avatar requires the cooperation between the agent domain and the region domain controlling the system with authority for the target virtual location. The quality of the system describing this interaction is "avatar presence."

3.2.1. Establishing Presence with the Region Domain

TOC

Once authenticated with the agent domain, the client application has established "account presence." Once in possession of a valid seed capability, the client application may request a set of capabilities representing services offered by the agent domain: digital asset management, instant message and voice chat support as well as placing the user's avatar into the virtual world.

Placing an avatar in the virtual world begins with the client exercising the "place my avatar in a region" capability. As part of this transaction, the client provides the URI representing a region. Upon receipt of this request, the agent domain determines the validity of the URL provided, and if the URL resolves to a trusted region domain begins the protocol between the agent domain and the region domain to place the user's avatar in the region.

The precise exchange of messages between each party is beyond the scope of this document, but is described in the OGP Teleport specification But a few important points should be noted:

* The protocol endpoint at the agent domain the client application uses to place the user's avatar in a region is provided to the client as a capability following successful authentication. It is not a publicly defined, fixed URL.

* The region the client wishes the agent domain to place their avatar in is represented as a URI. This URI may be a URN, in which case the agent domain SHOULD have the ability to convert the URN into a URL. If the target region is identified by a URL, it MUST use the HTTP (Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, “Hypertext Transfer Protocol -- HTTP/1.1,” June 1999.) [RFC2616] or HTTPS (Khare, R. and S. Lawrence, “Upgrading to TLS Within HTTP/1.1,” May 2000.) [RFC2817] URI schemes.

* The agent domain MAY apply a local policy to the URI and reject the request before attempting to connect with the region domain. (a "behind the firewall" agent domain may limit clients connecting to it to systems known to be inside the local intranet, for instance.)

* The agent domain MAY apply a local policy and reject the request after it makes an initial communication request with the remote region. (for example, if the region domain is operating servers with expired TLS certificates, or if those certificates are issued by a certifying authority the agent domain does not trust, it may reject the request.)

*The process of placing the avatar in the region results in capabilities from the region being communicated back to the agent domain for controlling the avatar. The agent domain SHOULD forward these capabilities to the client application.

*The process also results in the agent domain issuing capabilities to the region domain, allowing it limited access to information about the avatar such as the avatar's shape and appearance.

After an avatar is "placed" in a region, the agent domain is responsible for maintaining it's presence. That is to say, after the avatar has been successfully been placed in the region, the agent domain MUST refuse to allow a second region to "take" the avatar's presence without removing the avatar from its current region.

3.2.2. Moving Presence

TOC

When an avatar moves between regions, special care must be taken that the agent domain and both the source and destination regions end the process with the same understanding as to the avatar's location. Moving between regions is typically initiated by the client. The process is largely the same as the initial avatar placement, but with the important added step of removing the avatar from it's source location before rezzing it in it's destination. (In fact, the initial placement of an avatar can be thought of as a transfer from "nowhere.") The process of moving between regions is described in the OGP Teleport specification, thought implementers should keep the following important considerations in mind:

*The client signals to the agent domain it's desire to move from one region to another by accessing the same capability as is used for initial placement of the avatar.

*The agent domain must again check that local policy allows movement to the new destination, and MUST receive a capability for placing the client into the new region before it removes the avatar from it's current location.

*The agent domain MUST also remove the avatar from it's current location before placing the avatar in the destination location. Capabilities granted to the current region MUST be revoked as part of this process.

*The location of the avatar MUST be unambiguous and the agent domain MUST NOT represent the avatars location as being in two places at once. If required, for the short period between

removing the avatar from one region and placing it in another, the avatar's location may be "in transit."

3.3. User and Group Messaging

TOC

3.3.1. Spatial Messaging

TOC

Besides the presence of a fully articulated 3-dimensional representation of the user, the most important feature of the virtual world is interaction. The virtual world is a social space; communication with other users is important. Because the virtual world simulates features of consensus reality, "proximity chat" or "spatial messaging" is an important function. This mode of interaction allows users to "hear" text messages that are spatially proximal to the user's avatar, while ignoring other messages. The assumption being that avatar's whose users share a common interest will congregate in specific locations in the virtual world. Or they may find their avatars in the company of other users' avatars who are engaging in interesting conversation. Either use case is possible; emulating the consensus reality feature that people can hear conversations close to them, but not hear more distant conversations is an important feature of the virtual world.

Spatial messaging is managed by the region domain, and may be initiated by users' client applications or by the region itself. It is associated with an object in the virtual world (either an avatar or a "plain" object) and occurs at a particular location. The host in the region domain responsible for managing spatial chat applies a proximity algorithm to the chat to determine which avatars or objects are close enough to hear it. Those objects are all sent messages with the contents of the message.

Client initiated chat begins when the client application posts a message to the capability created by the region for an avatar's outgoing chat messages. This capability is given to the client after successfully establishing presence in the region. Incoming spatial chat messages are posted to the event queue established between the client and the region.

Complicating matters somewhat, spatial chat may occur near region boundaries. When this occurs, the host managing a region's messaging must have a mechanism to communicate chat messages to it's peers. Hosts responsible for spatial chat in a region must establish event queues with their peers in order to receive chat messages that originated near the region's borders.

3.3.2. User to User and User to Group Messaging

TOC

Instead of speaking on the "public" spatial chat channel (remember, each avatar within a defined range will be able to hear these chat messages,) users may send private user to user messages. These messages are managed by the user's agent domain. After authentication, a client may request a capability for establishing a instant messaging sessions. The client then accesses this capability, providing a unique identifier for the target user. If the agent domain is able to successfully establish a session with the target user, the message originator is provided a capability to which outgoing messages are posted.
User to Group messaging is similar, but groups are used as the target for a message.
Incoming user to user or user to group messages will arrive in the event queue shared by the client application and the agent domain.

3.4. Digital Asset Access and Manipulation

TOC

The virtual world contains multiple digital objects; they have a position and an orientation as well as a shape and potentially a texture and other features applied to them. OGP defines formats for describing objects and avatar shapes, but more importantly it describes the mechanism by which those digital asset descriptions are transferred between client applications, agent domains and region domains. OGP also defines a trust model and a basic permissions system, describing which users or groups have the ability to make changes to any given object.
Digital assets may be "at rest" or "in world." Objects "at rest" exist only as a description of the object, maintained by a network addressable server and accessible via a unique URL. When an object is "rezzed in world," its representation is transferred to a simulation host in a region domain and it becomes viewable by avatars and other objects in that region.
Several classes of digital assets are defined: primitive shapes, textures, sound and animations for example. In addition to the data describing the asset, metadata my be applied to objects. Unique identifiers for creators, owners and affiliated groups may be maintained by an object. Permission metadata may be added to an object to limit it's distribution to remote systems or to define the allowable operations by given users or classes of users. Object name, description and tag values may be applied and should help with indexing and searching for objects. Creation and modification dates may be applied to assist systems that cache assets. Recent discussions regarding open content licenses implies an interest in license metadata. Such metadata

could be of use to consumers of digital assets; allowing them to more clearly interpret the creators intent with respect to sharing.

3.4.1. Manipulating Digital Assets

TOC

A number of useful manipulations of digital assets "at rest" are defined by OGP. Where appropriate, asset metadata may be altered by directly communicating with the network host with authority for that asset. This host may be part of the user's agent domain, or in the case of region-specific assets, it could be associated with a region domain. It is important to note, however, that not all metadata is modifiable by all users, even the asset's owner. Specifically, the semantics of the creator metadata do not allow the owner to change the creator's identity. Group membership may carry some rights like the ability to manipulate the size, shape and texture of an asset, but not an asset's owner.

The ability to access or manipulate digital assets is based on the accessor's identity. Accessing and manipulating digital assets is performed via capabilities which expose the state of the asset to an authorized client. This requires positive identification of the accessor prior to access. In the case where an asset server is owned by the same authority as the agent domain, this access may be as simple as providing the proper capability after user authentication. In cases where the asset server is owned by a different authority, systems for deferred authentication may be necessary. Work is currently underway to integrate OAuth and SAML into OGP for this purpose.

At a gross level, the types of resources exposed by a digital asset server would include:

*a resource for searching an agent's inventory

*a resource for iterating across an agent's inventory

*a resource for accessing or manipulating a digital asset's metadata

*a resource for uploading new digital assets, or changes to an existing asset.

*a resource for removing a digital asset from the authority of the asset server's domain

*a resource for transferring the asset or a copy of the asset to a remote asset server

*a resource for instantiating an object "in world"

3.4.2. Establishing Presence for Digital Assets

TOC

Digital assets are intended to be used "in world," meaning there must be a way for a user to direct a simulation host to take an asset from an asset store and imbue it with presence in the virtual world. The separation between agent based services and region based services is fundamental to OGP and implies the authority for the system maintaining the asset "at rest" may be distinct from that which simulates the asset "in world." In practical terms, a region simulator may need to communicate with an asset server owned by a different person or company. In situations like this, trust is paramount. Because an asset's metadata may limit the owner's right to make copies of an asset, the agent domain MUST be able to trust the region domain will honor that metadata.

There are two levels of trust defined when working with digital assets: host-based trust and user-based trust. The former represents one system's expectation that the other will honor the metadata regarding ownership, creatorship and rights and restrictions implied by these concepts. Host based trust is carried by X.509 / PKIX certificates and implies a managed PKI. User-based trust represents the expectation the asset server will expose sensitive resources only to users with the right to access such resources.

Provided trust is established between the asset server and a simulation host, and the simulation host can demonstrate it is acting on behalf of a user with rights to access a particular resource, OGP defines a protocol for transferring a representation of the digital asset for simulation. As part of this protocol, access to a digital asset may be restricted while the object exists "in world." This is the case for objects whose creators or owners specify that only one copy of the asset may exist at a time.

4. IANA Considerations

TOC

This memo includes no request to IANA.

5. Security Considerations

TOC

As mentioned previously, the concept of a persistent, ubiquitous identity in the virtual world is core to the user experience. Keeping agent based services distinct from region or object based services has

advantages for scalability and flexibility. However, it does have ramifications for the security of the virtual world as a whole.
Most notably, this structure puts the agent domain in the role of a trust broker. That is, the agent domain is trusted both by the set of users who operate client applications and by the set of users who administer peer domains. A transitive trust relationship exists between the peer domains and end users by way of the agent domain. The administrators of the peer domain trusts the agent domain to properly identify end users, and potentially to ensure they are members of a particular class. The end users trust the agent domain to properly identify peer domains and to potentially limit the transfer of digital assets to only those domains that have explicitly agreed to honor asset permissions meta-data.
OGP does not REQUIRE domains to adhere to any preset policy, however. It instead provides a mechanism for communicating identity information so that such a policy MAY be enforced.

5.1. Capabilities

TOC

OGP makes extensive use of RESTful resources accessed via HTTP. Application state is communicated and changed by accessing web based resources. One characteristic of such resources is they have a well defined URL, many of which are formatted as URL-based capabilities. [I-D.lentczner-ogp-base] (Lentczner, M., “Open Grid Protocol: Foundation,” March 2009.) Capabilities have the characteristic that possession of the URL implies the right to access the resource it identifies. It is important that capability URLs are shared only with trusted participants. The OGP Base document defines the characteristics of URL-based capabilities, including the requirement that they include an unpredictable random component in the URL. Implementers need also ensure that these URLs are protected using suitable mechanisms (such as TLS, IPSec or link encryption.)

5.2. User Authentication

TOC

Prior to granting an end user access to any agent domain managed sensitive resource, the agent domain MUST authenticate the end user. The OGP Authentication specification defines three techniques for using shared secrets to authenticate end users. The agent_login resource used for end user authentication provides an extensible mechanism, allowing the development and use of additional authentication techniques (SRP, TLS Client Certificates, SASL, etc.)
Again, it should be noted that OGP as currently defined does not REQUIRE an agent domain to support a particular authentication scheme

(shared secret, public key, secure remote password, etc.) But it does define the mechanism for three shared secret options.
Once a user is successfully authenticated, their client application is passed a seed capability (as described in the OGP Base specification.) This seed capability is used by the client application to request access to resources and services managed by the agent domain (including services like "place my avatar in the virtual world.")

5.3. Agent Domain to Region Domain Authentication

TOC

Agent Domain authentication, or the process of authenticating an agent host to a region host uses a X.509 PKI. Prior to communicating, the agent domain generates a key pair for a particular agent host under their control and requests a certificate from each the region domain with which they wish to interact. The region domain returns a signed certificate to the agent domain which the agent domain uses in subsequent communication with the region.

5.4. Access Control for Digital Assets

TOC

In addition to security characteristics addressing traditional network and user security issues, the raison d'être of OGP is to communicate state concerning items inhabiting a virtual world. Some of these items may have access control restrictions within the scope of the applications used to simulate and render the virtual world. OGP defines an extensible permissions model which allows permissions meta-data to be associated with virtual items.

---

6. References

TOC

6.1. Normative References

TOC

[I-D.hamrick-llsd]	Brashears, A., Hamrick, M., and M. Lentczner, “Linden Lab Structured Data,” draft-hamrick-llsd-00 (work in progress), February 2009 (TXT).
[I-D.hamrick-ogp-auth]	Chu, T., Hamrick, M., and M. Lentczner, “Open Grid Protocol: Authentication,” draft-hamrick-ogp-auth-00 (work in progress), March 2009 (TXT).

[I-D.lentczner-ogp-base]	Lentczner, M., “Open Grid Protocol: Foundation,” draft-lentczner-ogp-base-00 (work in progress), March 2009 (TXT).
[RFC2119]	Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).

6.2. Informative References

TOC

[I-D.saintandre-rfc3920bis]	Saint-Andre, P., “Extensible Messaging and Presence Protocol (XMPP): Core,” draft-saintandre-rfc3920bis-09 (work in progress), March 2009 (TXT).
[RFC1822]	Lowe, J., “A Grant of Rights to Use a Specific IBM patent with Photuris,” RFC 1822, August 1995 (TXT).
[RFC2616]	Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, “Hypertext Transfer Protocol -- HTTP/1.1,” RFC 2616, June 1999 (TXT, PS, PDF, HTML, XML).
[RFC2817]	Khare, R. and S. Lawrence, “Upgrading to TLS Within HTTP/1.1,” RFC 2817, May 2000 (TXT).
[RFC3986]	Berners-Lee, T., Fielding, R., and L. Masinter, “Uniform Resource Identifier (URI): Generic Syntax,” STD 66, RFC 3986, January 2005 (TXT, HTML, XML).

Appendix A. Definitions of Important Terms

TOC

agent domain
The agent domain is the administrative authority responsible for managing services related to avatars and users. Identity management, group membership, avatar appearance, profile information, user authentication and group messaging are examples of services and information maintained by the agent domain.

agent host
A network host maintained by the agent domain is called an "agent host."

avatar
The avatar is the representation of a user in the virtual world. The avatar's shape and appearance are used by other users to render a graphical representation of the inhabited virtual world.

The user's view of the virtual world is rendered from the perspective of their avatar.

client application
A client application is any application that is operated for the benefit of the user. Common client applications might include a "viewer" that renders the virtual world on the user's workstation or a web application used to manipulate the user's digital assets. OGP does not provide a canonical list of client application categories, but if an application is not a part of an agent domain or a region domain and it is manipulating user data or an avatar on behalf of a user, with the user's permission, it is a client application.

region domain
The region domain is the administrative authority responsible for managing services related to presence in the virtual world and it's simulation. Typical services exposed by a region domain would include physics simulation, avatar presence and virtual object presence lifecycle management (i.e. - the creation, manipulation and destruction of objects in the virtual world.)

region host
A network host maintained by the region domain is called a "region host", though the historical term "simulator" is still very common.

user
The entity controlling an avatar in world is the "user".

Appendix B. Acknowledgements

TOC

The author gratefully acknowledges the contributions of Mark Lentczner whose architectural guidance was instrumental in the development of this document. And of David Levine whose insights and review contributed greatly.

Author's Address

TOC

	Meadhbh Siobhan Hamrick
	Linden Research, Inc.
	945 Battery St.
	San Francisco, CA 94111
	US

Phone:	+1 650 283 0344
Email:	infinity@lindenlab.com