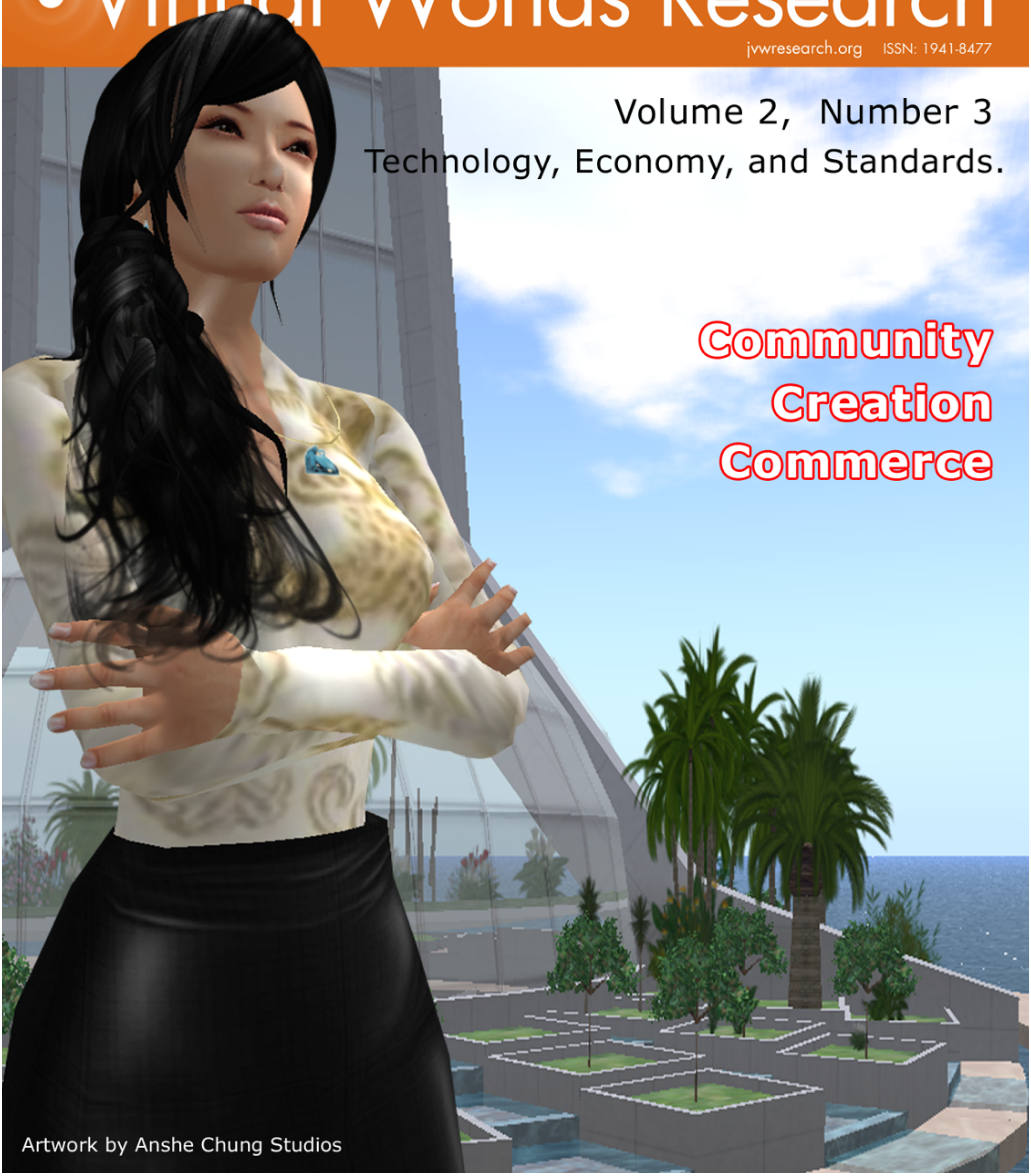
Journal of Virtual Worlds Research

jvwresearch.org ISSN: 1941-8477

Volume 2, Number 3

Technology, Economy, and Standards.

Community
Creation
Commerce



Artwork by Anshe Chung Studios

Volume 2, Number 3
Technology, Economy, and Standards
October 2009

Journal of Virtual Worlds Research
jvwresearch.org ISSN: 1941-8477

Volume 2, Number 3
Technology, Economy, and Standards
October 2009

Virtual World Interoperability:
Let Use Cases Drive Design

By Jon Watte, Forterra Systems

Abstract

This article examines the history of virtual world interoperability as evidenced through early systems like DIS and HLA, current systems such as Second Life / OpenSim teleport and OLIVE simulation interoperability, and the future, interconnected metaverse. The article argues that "serious" virtual worlds will be the initial market that drives true virtual world interoperability because of its particular needs. Based on this claim, a comprehensive approach to standards-based virtual world interoperability is described.

Keywords: virtual worlds; interoperability; metaverse.

Virtual World Interoperability:
Let Use Cases Drive Design

By Jon Watte, Forterra Systems

Virtual worlds are slowly creeping into our daily lives. While some early adopters have been using them for entertainment, research and training over the last 20 years, virtual trade shows and online conferencing with user avatars are putting them front and center on the desktops of workers around the world. However, while a "walled garden" virtual world may be useful in and of itself (just like a cell phone being able to call other cell phone customers using only the same carrier), the real usability explosion will come when the different virtual worlds start talking to each other, just like cell phones being able to call any phone number in the world, no matter who the destination carrier or operator is. In fact, telephony has grown into a large infrastructure used for conference calls, IP telephony, telefax, and even video calls in some parts of the world. This growth could not have happened without interoperability between systems, operators and technologies, where that interoperability allowed the main feature of the telephone (carrying a band-limited signal between two endpoints) to spread everywhere. By comparison, interoperability between virtual worlds, where such interoperability would only allow, for example, the same virtual currency to be used in different places, would not enable the same level of widespread use; the meat of a virtual world is its ability to support spatially based interaction between users, and between users and the simulated world.

Virtual Worlds in Context

It is important to be clear on the context within which a given argument is made. Without understanding and making clear the underlying assumptions and history of an argument, the argument can easily be misunderstood or simply not appear relevant. In order to mitigate that problem, I will describe the context of virtual worlds used in this article. I will start by narrowing down the definition of the kind of virtual world I want to discuss.

In discussion, social web sites, such as Facebook or LinkedIn, can arguably be classified as virtual worlds. After all, they provide interactivity, a meeting place for users, persistence of user-initiated changes and a rule set under which interactions are made – all of which are traits seen in most virtual worlds in use today. However, I argue that broadening the definition of virtual worlds to include 2-D web sites like Facebook is not meaningful, because the mode of interaction is very different from a 3-D virtual world like AlphaWorld, Project Entropia, or There.com. Any attempt to find commonality between these worlds will fall back to simple, web-based, transactional interactions, for which standards already exist or at least are emerging (technologies from EDI to OpenID to SOAP falls in this category).

Instead, to separate virtual worlds from web-based social spaces, I will focus on virtual worlds that include real-time, 3-D, physically based interaction between users. As any virtual world user will tell you, the real-time, 3-D, physically simulation-based interactivity is a major part of what makes a virtual world special. Human beings have evolved to have acute spatial awareness, and relate to objects in the environment in 3-D. VR research[i] from the 1980s and 1990s show that a physically based 3-D virtual world draws upon this awareness in a way that flat services cannot, and thus deliver more immersion and a sense of presence.

Another distinguishing factor of virtual worlds is the ability of users to modify the environment in a persistent way. Unlike 3-D games, like World of Warcraft, Counterstrike, or EVE Online, a virtual world allows users to make permanent modifications to the environment and objects in the world and generally to introduce artifacts that change the simulation of the world more or less permanently. For example, in Second Life, a user can create a new object and attach a script that flings any user that stands on the object into the air – in effect introducing a user catapult. Because of this, the main attraction of a virtual world for entertainment is the content that the users can create themselves – be it a virtual mansion, night club, or Rube Goldberg-style contraption. By contrast, a 3-D game, even though it may feature thousands of users in a physically simulated 3-D world, does not generally allow persistent modification of the world by players.

Within this paper, I will separate two sub-categories of virtual worlds, the usage of which differs sufficiently to warrant such separation. In an "entertainment" virtual world, users attend the world in order to enjoy themselves. The entertainment virtual world is a destination or mode for an experience, much like a movie theater is a destination for an experience, or a phone call to a friend is a mode for an experience. Meanwhile, "serious" virtual worlds are made to achieve specific goals related to training, education, collaboration, or other day-to-day work-based interaction. In this case, it is not the experience that is the main take-away; it is the outcome of the collaboration (lessons learned, meeting deliverables, etc). From a market point of view, an entertainment virtual world may compete with a real-life bar or night club, or perhaps watching TV, whereas a "serious" virtual world competes with a classroom, a conference call, or an in-person meeting.

One formulation of the difference between 3-D virtual worlds and other online interactive or semi-interactive technologies is the concept of "3D3C," although in that formulation, the third "C" (in-world Commerce) is more a requirement for current entertainment worlds than for current business worlds because of the different usage modes. The two other "C"s map well to both entertainment worlds and serious worlds, where Community is the users who interact, and Creation is the interaction with the environment, and the actual work being done.

Previous Virtual World Interoperability

To better understand where we want to go, it is useful to understand where we've been and what we've learned so far. This necessitates a brief overview of the capabilities and technologies used for virtual world interoperability so far.

For the past five years, my work at Forterra Systems has involved interoperating between our enterprise virtual world platform OLIVE and a number of other systems. At the same time, our licensee Makena Technologies operates the entertainment virtual world There.com, giving us a good view of the needs and desires of entertainment users and operators. Based on this experience, as well as following the market in general, I have concluded that entertainment virtual worlds do not have a huge demand for interoperability from the end users. This is important, because in the end, if there are no users willing to drive and fund interoperability work, then such work is unlikely to be successful. To put it another way: When asked "how much would you pay to be able to teleport from There.com to Second Life and back again, without switching client applications," the overwhelming majority of users would answer "not much."

By contrast, all of the enterprise virtual world integrations we have made so far have incorporated some form of interoperability. That interoperability may be simple, such as authenticating users to an existing LDAP database, or providing the ability to call into and out of the public telephone network (typically using a SIP gateway), or complex, such as the ability to plug in a third party physiology model to simulate the health of avatars when running exercises for medical training. When enterprise customers are asked how much they are willing to pay for interoperability, the answer is generally “it’s a crucial requirement.”

From this experience, I have learned that the main area of interoperability need that is underserved for virtual worlds is the interoperability of entities, where “entities” are defined as objects that generate forces or interactions in the world – avatars, vehicles, communications equipment, etc. By contrast, non-entity objects in the world would be “dumb” objects, such as rocks, trees, buildings, and others. While a rock may fall and tumble based on gravity and collision, it does not introduce any behaviors of its own into the world.

For entity interoperability, we have had great success using the DIS protocol (IEEE 1278[ii]). This protocol grew out of the work that the United States Department of Defense (DOD) did in the 1970s and 1980s with regards to military simulation interoperability.

The need, at the time, was to couple different simulators (for systems like army tanks, airplanes, ships, and satellites) together, so that the operator of a flight simulator could see friendly and enemy tanks on the ground, and even interact with them (mainly through weapons systems and sensors). In this model, each simulated system (each individual tank or airplane) was its own simulator, receiving telemetry from all the other simulators, and using dead reckoning to interpolate the position of those entities between updates. With dead reckoning, periodic updates of the state of an entity are forward extrapolated by the receiving end to calculate how an object is likely to evolve over time. For example, if I know that you were at a certain position 100 milliseconds ago, and I knew your velocity at that time, I can make a pretty good guess at what your position is now by adding that velocity, times 100 milliseconds, to the old position. Dead reckoning allows objects to be displayed in a consistent time frame of reference, but instead trades off accuracy – from the time you make a turn, until the time that a network message gets to me telling me you made that turn, I will still assume that you are moving forward. The alternative is to display objects using past state, and only update the state of objects as new updates are received. In highly kinetic activities, such time delay may be much less desirable than the spatial inaccuracy introduced by the “guessing” of dead reckoning[iii].

As technology progressed and computer capability increased, a kind of system known as Semi-Autonomous Forces (SAF) gained in prominence. This kind of system uses algorithms to simulate the behavior of entities of various scales, from an individual dismounted soldier, through platforms like vehicles and ships, all the way up to aggregate entities like battalions. DIS was modified to support the introduction of SAF into a simulation, so that some of the entities would be driven by user-operated simulators, and other entities would exist only as virtual entities inside the SAF constructive simulation. At the same time, real-time telemetry, made possible through better instrumentation, GPS systems and other technological advances, could be linked into a simulation, providing a virtual view of real world entities such as airplanes and vehicles. When the simulated entities are fed back into the real world entities’ display systems, such as heads-up displays in a cockpit, the full integration of Live, Virtual, and Constructive simulation is achieved. All of this has been done with the DIS protocol, which has proven to be very robust, and a good vehicle for interoperability between very different kinds of systems.

In the 1990s, the DOD started building a new simulation interoperability standard known as the High-Level Architecture (HLA), which later was standardized as IEEE 1516[iv]. Unfortunately, this standard was more concerned with things like supporting constructive and event-based simulation at non-real-time pace, thus supporting vendor-specific solutions to the problem of distributing time management into the GALT[v] (Greatest Available Local Time), rather than defining any goal of “plug-and-play” interoperability between disparate systems. In the end, HLA is an API specification, not a wire protocol, and thus, two simulators that want to interoperate have to use the same API implementation. API implementations are commercially available from vendors like MAK Technologies, Pitch, or the large system integrators. Additionally, HLA allows each simulation to define its own object model, using a text-based format describing the FOM (Federation Object Model) to use for the simulation. All in all, this means that hooking up two separate simulators with HLA requires significantly more work than hooking them up using DIS, because DIS is a wire protocol with well-define object model, whereas HLA requires re-linking (and in some cases re-compiling) as well as FOM mapping to work right. For those of us mostly interested in real-time interoperability, it is generally understood among many practitioners that HLA does not meet the interoperability requirements as well as DIS does in practice. While we at Forterra have made sure that our system can interoperate using HLA, no customer of ours has yet actually used that particular technology.

Since 2005, the On-Line Interactive Virtual Environment (OLIVE) from Forterra Systems has been able to participate in a DIS simulation, exchanging vehicle, avatar and fire/explosion data with live, virtual and constructive simulators inside the DOD. It is even possible to join two separate OLIVE systems (or other virtual worlds using DIS, if they were available) into the same simulation, achieving a high degree of interactive interoperability between different virtual worlds. This positive experience suggests a fruitful way forward for future virtual world interoperability, which I will discuss below.

Almost every installation of OLIVE now comes with some sort of interoperability, and the main form of interoperability requested is where multiple systems are merged together to form a “super-system,” that integrates the capabilities of all systems into a richer capability, affording users the benefits of all the systems that are integrated. Organizations as diverse as Accenture Consulting, InWorld Solutions, and ACS are finding that virtual worlds are often more effective than traditional means of meeting and collaborating (such as conference calls or video conferencing), and often can deliver something close to the experience of an in-person meeting at a fraction of the cost. Often, the cost is even lower than the cost of a phone conference!

Teleporting Between Worlds: A Detour

In 2008, OpenSim open source virtual world project members showed a demo, where they teleported avatars from a Second Life simulator instance to an OpenSim simulator instance. Unfortunately, the assets involved in representing the avatars were not available at the destination, so all the avatars ended up with the default look. Before transportation, users of the OpenSim simulator could not see the users that were in the Second Life simulator; after transportation, users of the Second Life simulator could not see the users that moved to the

OpenSim simulator. Further, the client from within which teleportation was done used the Second Life client/server protocol, and the source and destination servers both used the Second Life scripting and geometry system – the main thing that was transported between the two systems was the identity, using an identity authentication system similar to the available OpenID protocol, and a hand-off between servers where one client was instructed to disconnect from one server and connect to another server.

While an interesting experiment, the value of the capability is currently low. Interoperability that demands that all parties use the same simulation, networking and rendering technology at a low level, is no more interoperability than cell phones that can only call other phones using the same wireless technology. Further, even had the teleport included the details of the avatars (look, behavior, and other details), it's unclear what the added value is worth, compared to the users just logging out using the Second Life client, and logging on using the OpenSim client – interoperability, in that guise, is a convenience that saves the user some hassle, but does not deliver any new capabilities compared to a "parallel" or "side-by-side" situation. This is in stark contrast to the very real, additional capabilities that protocols like DIS have already delivered to virtual simulations for 20 years or more. Thus, it is reasonable to believe that more value will be delivered to users if interoperability involving multiple systems at the same time is achieved, than if simple "browser" interoperability is achieved because in the "browser" model, only a single virtual world can participate at a time. Leaving one world, means that you leave all the capabilities of that world behind, and take on different capabilities in the new world. It would be more desirable if you could merge the two worlds, in effect providing some form of union of the capabilities of both the worlds.

Use Cases: A Way to Focus

Given that entertainment use of virtual worlds is largely focused on experiences, usually created by other users, the benefits of interoperability between different virtual worlds in that area seem diffusely understood at best. Meanwhile, there is a clear need for interoperability in the world of enterprise and "serious" virtual worlds, where merging systems together creates clear benefit to business users. Thus, it stands to reason that one driver of virtual world interoperability will be just these business users, trying to merge systems together to create a better tool for getting their core job done. Against this background, I have extracted five separate use cases, which I describe in some detail below. It is my hope that these use cases will contribute some focus to the global discussion of virtual world interoperability, and provide food for thought when standards bodies like IEEE, IETF, or MPEG start considering the needs of virtual worlds.

Use Case 1: Friend Invite

1. User A uses virtual world system A that complies with simulation interoperability standards.
2. User B uses virtual world system B that complies with simulation interoperability standards.
3. User A wants user B to visit him/her in system/world A, and gets a suitable URL from his/her system (A), which he/she sends this to user B using any transport (through either mail, IM, integrated communication, carrier pigeon, and more).
4. User B clicks/activates this link in a browser, e-mail client, or similar.

5. After a brief "loading" screen, user B sees user A in user A's environment, including a representative form of any simulated object in that environment.
6. User B can interact at some level with the objects from user A.
7. Objects that user B take out of inventory show up in some representative form for both user A and user B.
8. User A can interact at some level with any objects that user B bring out of inventory.

The benefit of this use case is that users of different virtual worlds can invite and communicate with each other using the virtual world metaphor, regardless of the particular virtual world technology used for their "home base" virtual world, presumably purchased and supplied by their employer or one of many third party virtual world service providers.

Use Case 2: Collaborative Training

1. Company A operates a chemical plant in city B. Company A uses virtual world system A to do simulation/training/command-and-control of its plant.
2. City B has an emergency response organization that uses virtual world system B for training and scenario planning.
3. At a defined time, company A and city B agree to connect their worlds for a defined duration to conduct a training exercise related to a fire in the chemical plant.
4. At the defined time, a representation of the detailed model/simulation of the chemical plant shows up at the right addressing the virtual world for the city workers.
5. At the defined time, city workers (ambulances, fire trucks, and others) become visible to the chemical plant workers.
6. Interactions between users of the systems include conversations (voice, simulated radio, PSTN).
7. Interactions between users of the systems include a display of the fire as it propagates based on company A simulation models.
8. Interactions between users of the systems include the ability for firefighters to pour water (or other agents) onto the fire, and have the simulation respond.
9. Interactions between users of the systems include the ability for city workers to load a chemical plant worker into a city ambulance.
10. At the pre-determined time, the interoperability ends; the city disappears from the company plant, and the company plant disappears from the city model.
11. Session record/review capability used by the city in virtual world B includes all communications and interactions made in the system including those internal to company/world A.

The benefit of this use case, in addition to the Friend Invite use case, is that interoperability can be limited in time and (virtual) space to protect potentially sensitive information. Additionally, this use case shows the benefit of defining interactions between objects operated by one system with objects operated by another system, leading to synergistic simulation similar to that evidenced by the DIS protocol, but applicable to a broader, non-military audience.

Use Case 3: Scene Transfer

1. A user of virtual world A has prototyped an interesting environment.
2. The user decides to donate that prototype to an organization that uses virtual world system B.
3. The user "exports" his/her prototype to a series of common data containers (textures, meshes, scripts, and others) of some standard format (e.g., COLLADA, X, FLT).
4. All content that the user has created and owns the rights to that is part of the prototype is included in sufficient detail in the export.
5. All content that is part of the prototype and that A has exposed sufficient permission for is included in sufficient detail in the export.
6. No content that is restricted from this kind of use is included in the export, although a reference saying "an object with characteristics C named N was here" may be.
7. The exported data is attributed (in aggregate) to the user of world A.
8. Organization B can load the exported assets into their virtual world.
9. Meshes and textures in a well-known standard format shows up in world B as expected, with attribution to the user from world A, no matter what technology the respective virtual worlds use.
10. Scripting and interactive behavior shows up only if the destination virtual world implements a scripting or behavior system compatible with the source world.

The benefit of this use case is that work done to develop virtual world content for one world can be transported to another world with minimal manual intervention. While things like scripting and simulation algorithms may not transfer over (depending on the degree of implementation similarity between source and destination), the main 3-D content, including meshes, textures, and layout, does. Additionally, such transfer is shown to respect intellectual property rights of content that may have been re-used to generate the scene.

Use Case 4: Analysis

1. ISV (Independent Software Vendor) A creates a system for analyzing movement of avatars in a virtual world.
2. The product from ISV A can be connected to any virtual world or worlds implementing interoperability.
3. When the tool is connected, certain patterns of movement are detected and flagged by the tool.
4. The tool can report recognized actions through chat, or through introducing "flag" objects into the world.
5. A virtual world user interacting with the "flag" objects can pull up a web page that gives information about the detected interaction.

The benefit is that development effort to generate various tools can be replicated across multiple virtual worlds, saving a lot of re-implementation effort for ISVs interfacing with the virtual worlds market. Additionally, the benefit of a commonly agreed external data representation enables formulation of standardized metrics and measurements, which is expected to greatly help research into the use and evolution of virtual worlds.

Use Case 5: Data Logger

1. User of virtual world system A purchases a 'data logger' tool from company B.
2. When attaching the data logger tool to the virtual world, the data logger receives information about all the objects, interactions, and communication in the system.
3. After the logger has been detached, the data logger tool can be seen as a separate virtual world peer, similar to any external virtual world server, and can be connected to by any virtual world using interoperability, with the difference that the simulation can only be experienced, not impacted.
4. The logger implements play and shuttle controls that allow the action from the original session to be re-played at a later time. Any attached virtual world peer will see the recorded actions.
5. Enough data is available to the logger that search functions like "find the time when avatar X interacted with vehicle Y" can be implemented.
6. Actions by avatars in the connected peers during playback do not affect the objects provided by the logger tool.

A generally agreed-upon data presentation and interchange standard, implemented using server peer-to-peer co-simulation of a shared space, enables a large variety of use cases. The Data Logger is interesting in that it shows how data can be both consumed and produced by systems that are not in themselves virtual worlds, yet provide clear benefits to users of virtual worlds. Like use Case 4: Analysis, the ability to do this with any world significantly reduces the burden on ISVs. Additionally, one can consider the potential future markets that open up when virtual world record and playback (in full 3-D, as opposed to a plain video stream) is a deployed, easy-to-use, generally applicable capability.

Conclusions

These use cases cover a large part of the kinds of interoperability that we have seen customers of virtual worlds ask for and that we have so far delivered either using proprietary technology, or using existing domain specific protocols and methods. Based on that experience, I believe that the most worthwhile interoperability approach for virtual worlds is to solve the problem of tying the worlds themselves together. Some problems that will come up are already pretty well solved, often more than once:

- How to describe where static data is located and retrieve it: HTTP URIs are ubiquitous and have lots of available infrastructure.
- The format of graphical data and meshes: COLLADA is widely supported. Also, formats like OpenFlight and X3D also are open and have some support in parts of the industry.
- How to support authentication across domains: OpenID / OAuth already serve as the de-facto authentication system for Web 2.0 systems.
- Streaming voice communications: SIP is used extensively for VoIP solutions, and supports conferencing. It would have to be extended with 3-D entity reference information to provide a full solution, but it solves a large part of the problem already.

The overwhelming majority of virtual worlds systems today use the client/server model, with servers being authoritative over important state of objects in the simulated world, and clients serving to present the simulated state to users, and accept further input to affect the world simulation, and communications for other users of the virtual world. Based on the success of protocols like DIS, where a wire protocol and defined object model allows true plug-and-play interoperability between different simulation systems and technologies, I believe that a similar model, adapted to the realities of the Internet and the domain of current virtual worlds, would be highly successful. Such a model could also scale to peer-to-peer systems just like DIS is used peer-to-peer: each peer is considered a participating node in the system.

To interoperate, two systems would have to agree on some shared space where the simulation will take place. This space is populated with terrain (static geometry and obstructions) based on a shared understanding (such as a common scene/database), and each system can then introduce entities (such as avatars) into this space. Each class of entity (avatar, furniture, vehicle, and more) could support standard interactions, such as walking to a location, or sitting an avatar down. Similarly, the presentation of entities (sound, textures, meshes, and animation) could be shared using standard file formats like COLLADA, DDS, and MP3. Because each system already knows how to present its own entities to its own users, the main work involved would be translating the entity information from other systems such that the connected clients could display them.

When examined in more detail, it turns out that this model has a number of implementation benefits, too. Because each system is free to implement the client/server communication in any way it wants, a “lowest common denominator” protocol does not need to limit what can happen in any one virtual world. Because each world simulates its own entities and provides mainly presentation information and results (interactions) of that simulation to the other systems, the systems do not need to agree on how physics, scripting, or other simulation mechanics are implemented. As long as the outcome of the simulation can be made visible to other systems within the shared space, heterogeneous systems can collaborate to create a “shared illusion” of the shared space.

On the practical side, implementing a common wire protocol and object model for interoperability and allowing servers to translate to their own clients, is also estimated at an order of magnitude less work than other options, such as trying to move responsibility for a simulated entity between different systems – doing so would require that all systems understood the same low-level simulation, scripting and logic functions, which is not a reasonable assumption in a competitive landscape, and could significantly curtail both innovation and implementation flexibility.

Thus, if you care about interoperability between virtual worlds, I hope that you will give these arguments serious thought, and join me in the work to create a future, interconnected, vendor and technology inclusive metaverse based on simulation interoperability.

---

[i] Schuemie et al, 2001: Research on Presence in VR: a Survey; Journal of Cyberpsychology and Behavior
[ii] http://ieeexplore.ieee.org/servlet/opac?punumber=2826
[iii] Brook Conner and Loring Holden, 1997: Providing a Low Latency User Experience in a High Latency Application; Proceedings of the 1997 Symposium on Interactive 3D Graphics
[iv] http://ieeexplore.ieee.org/servlet/opac?punumber=7303
[v] Hong et al, 2007: Design and implementation of time management service for IEEE 1516 HLA/RTI; Proceedings of the 2007 summer computer simulation conference