to appear in
Proceedings of the Eighth IEEE Distributed Simulation – Real Time Applications Workshop, 2004

An Architecture for Web Services Based Interest Management in Real Time Distributed Simulation

Katherine L. Morse
Ryan Brunton
SAIC San Diego
morsek@saic.com
bruntonr@saic.com

J. Mark Pullen
Priscilla McAndrews
George Mason University
mpullen@gmu.edu,
pmcandre@netlab.gmu.edu

Andreas Tolk
James Muguira
Old Dominion University
atolk@odu.edu,
muguira@computer.org

Abstract

The Experimentation Command and Control Interface (XC2I) project has developed an architecture for a Web-service based viewer/controller for use with distributed simulations supporting military experiments. As part of this activity, a capability for Interest Management with three functions is being created. The functions are Role-Based Access Control (RBAC), Area of Interest Management (AOIM), and Aggregation Interest Management (AGIM). While the approach is compatible with High Level Architecture (HLA) for Modeling and Simulation, primary information exchange takes place using the Web services, i.e. software-to-software messaging interfaces that operate over Web protocols such as XML/SOAP. This paper presents a Web Services Internet Management (WSIM) architecture designed to achieve these capabilities in a way that is compatible with simulations using the HLA. The protocols and information flow structure are described, along with the architecture's design rationale, interest management rules, and plans for its implementation. The paper concludes with a description of potential for future development of WSIM, including adoption of the Command and Control Information Exchange Data Model (C2IEDM) and use of Overlay Multicast for data distribution.

1. Introduction

The Extensible Modeling and Simulation Framework (XMSF) represents a new approach to interoperation of software to compose distributed simulations. XMSF entails use of emerging commercial Internet/Web technologies for message-based exchange of data, and was expounded in [1], based on the authors' work in the modeling and simulation technical community. The distinguishing characteristic of XMSF is use of the Extensible Markup Language (XML) to exchange messages using a common ontology cast as an XML tagset. The central architecture is that of a Web service, defined by the World-Wide Web Consortium (W3C) as:

> *A Web service is a software system identified by a URI [RFC 2396], whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols* [2].

Protocols in the Web services stack, depicted in Figure 1, are:

The *Extensible Markup Language (XML)* which describes the components of the web service, in particular the structure and content of input and output data.

The *Simple Object Access Protocol (SOAP)* which wraps XML in a form that can be used to invoke a remote service

The *Web Services Description Language (WSDL)* which supports definition of the function(s) of a Web service

The *Universal Distribution Discovery and Interoperability (UDDI)* protocol which supports registry processes for automatic composition of services.

Any standard Internet data transmission protocol; most commonly the *Hypertext Transport Protocol (HTTP)* is used, however other protocols such as the *Blocks-Extensible Exchange Protocol (BEEP)* and even the mundane *Simple Mail Transfer Protocol (SMTP)* can be used.

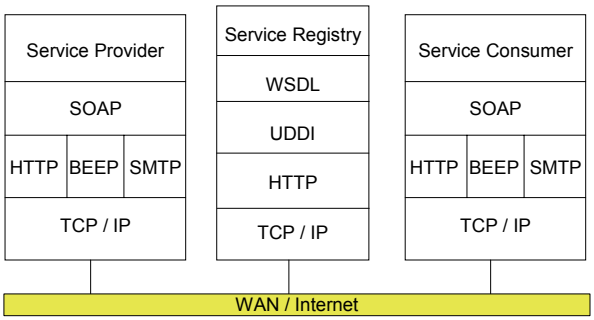

Figure 1. Web Services Protocol Stack

In [3], Pullen *et. al.* described how these protocols can be used to support distributed simulations under the High-Level Architecture (HLA). As XMSF grows into larger simulation environments, it will require capability to support both interest management, where data is transmitted only if there is a defined need for it, and streaming updates from real time simulations to networked viewers. Among the early successes of XMSF cited in [3] is the Extensible Distributed Continuous Experimentation Environment Viewer (XDV) created for the US Joint Forces Command (JFCOM). We are now engaged in expanding on the XDV concept to create a viewer-controller for a large-scale HLA federation used to support experiments in Joint Urban Operations (JUO). This project has required us to develop an architecture for Web Services-based Interest Management (WSIM), which we present below. The remainder of this paper is divided into sections that define interest management broadly, address the characteristics of Web services that must be considered in such an architecture, describe the architectural elements we have defined, and address special concerns for streaming data delivery in a Web service environment.

2. Interest Management Background

Interest management is defined in [4] as "limit[ing] the amount of information passed across a communications interface to the information of interest for a certain user perspective at that moment in time." Very large scale distributed simulations designed around broadcast based protocols suffer from two scalability issues with respect to network traffic[1]. The first is the sheer volume of data that is sent on the network. In some cases, this data may be irrelevant to all the other simulations, or at least to all the simulations on a subnet[2]. The second issue is the ability of the receiving simulation to process and discard irrelevant traffic fast enough to get to relevant traffic in a timely manner. Obviously this latter problem is of more concern in real time simulations.

The concept of Interest Management[3] was developed to address these problems by reducing the received messages to a smaller, relevant set [5]. Under Interest Management, a simulator expresses its data interests in terms of location and other application-specific attributes. The data interests are referred to as the area of interest (AOI) and usually correlate with the sensing capabilities of the system being modeled. For the different sensory modalities, the area of interest may have different sizes and shapes. The simulator maps its AOI into a (usually simplified) representation known as an interest expression (IE). Other components of the simulation infrastructure, interest managers (IMs), accept the simulators' IEs and use them to filter messages to sets (or reduced supersets) which meet the receivers' needs. The primary purposes of interest management are:

- Operationally, to support the user's need to define what information from the simulation will be displayed to increase situational awareness, and
- Technologically, to reduce the amount of network traffic, a serious problem in large-scale distributed simulations such as JUO.

Operational questions (what will be displayed and how it will be displayed to increase situational awareness) are beyond the scope of this paper. Here we will assume that the user's choice makes operational sense and therefore should be supported as well as the technology will allow. It follows that we must apply effective technical solutions to optimize delivery of only that data the human users and the supporting simulations need. The following considers interest management in that sense.

Among the best-known technologies for interest management is the Data Distribution Management (DDM) of the HLA [6, 7, 8]. DDM in the HLA takes the form of range specifications for generic dimensions, not necessarily in geometric or geographic terms. For example, HLA DDM supports definition of a spectrum of radio frequencies as a single dimension, so that the range specification covers a particular frequency band, as well as definition of 2-D or 3-D geographic regions where the dimensions might be based on either Cartesian or latitude/longitude coordinate systems. DDM itself is neutral regarding the definition or meaning of the dimensions.

However, it is our goal that WSIM not be limited to the functions supported by HLA DDM. XC2I is not limited to HLA compliant simulation systems or even to use in the modeling and simulation context; the same technology can be used to support common operating pictures rooted in heterogeneous command and control infrastructures. Therefore, we have taken a broader view

[1] After these two problems are solved, there may also be the issue of the receiving process having the capability to make meaningful use of the subscribed data.

[2] Multicast is the technology of choice for implementing many interest management schemes, especially between subnets.

[3] Interest Management also is referred to as relevance filtering, data distribution management, or data subscription.

of interest management in a Web environment. We see WSIM as having three broad components:

- A Role-Based Access Control (RBAC) component that specifies what information a particular user is entitled to access, on a role-based basis;
- An Area of Interest Management (AOIM) component that specifies, for any dimension defined by the simulation, what range of values the user wants to see; and
- An Aggregation Interest Management (AGIM) component that specifies, for any aggregation principle, how the detailed data in the simulation shall be displayed as aggregates; an example would be military forces displayed as combat units such as tank companies rather than individual weapon platforms such as tanks.

3. Web Service Opportunities and Constraints

The XDV project that preceded XC2I was based on the premise of creating a platform-independent and simulation-independent viewer, usable with commercial off-the-shelf (COTS) computer systems. XDV was based on open standards, using Internet protocols for communications. XC2I extends these principles to an operational viewer/controller for real-time distributed simulations, using Web services.

The Web Service vision is that services will work together seamlessly because they are developed to the same standards for self-description, publication, location, communication, invocation, and data exchange capabilities. As all the standards concerned are open, the technologies chosen for Web services are inherently neutral to compatibility issues that exist between programming languages, middleware solutions, and operating platforms. As a result, applications using Web services can dynamically locate and use necessary functionality, whether available locally or from across the Internet.

One great strength of Web services is that they are transaction-based. This confines the complexity of the data transfer model. A Web transaction may involve multiple tagged values, but it is atomic. However, the transaction-based model has its limitations. Each transaction incurs the full overhead of the request-response protocol and, therefore, generates many times as much network traffic, and requires many times as much processing, as a simple state update would. Also, currently there is no mechanism to subscribe for a "push" service; a client must "pull" the values it needs. (The W3C has a draft publish-subscribe mechanism in progress [9], however it is far from ready for standardization.) To overcome these limitations, we envision extending the basic transaction model to include a transaction that initiates delivery of a stream of real time data for object state attributes, such as geographic position, that are likely to change continuously.

The transaction-based model also provides an excellent way to approach access control. The client can obtain a token indicating that its user has access privileges to specific data. The token is then presented to the server as part of the transaction requesting the data. In the streaming case, direction of the stream, or the key to decode a broadcast stream, can be controlled in the same way.

Another facet of Web technologies with great potential to improve distributed simulation practices is the *registry* concept. Once a Web Service has been provided, there needs to be a mechanism to inform potential users (including distributed software processes) about the availability of the Service. One way to provide this information is via a directory Web service called a registry, where potential consumers can locate (discover) services meeting their requirements. UDDI is the standard for creating and utilizing global registries of businesses and business Web Services, where any company can locate and interact with the Web Services of from other providers. The same concept could apply to Web-based distributed simulation, if the registry approach is adopted there.

The next step, and a larger leap of faith, is the use of WSDL to compose distributed simulations in real time. In order to realize this vision, it will be necessary to achieve semantic interoperability based on thoroughly defined ontologies that can be matched during the process of composition to reach a conceptual level of interoperability. It is likely that a general theory of composability will be needed before this capability becomes a widespread reality.

A particular concern is the unambiguous information exchange needed for conceptual interoperability. XML provides for information exchange, but it does not guarantee that a given tag exists and has the same meaning in different systems. A tagset that is common among participating services is necessary to establish semantics consistency, whioch in turn is needed to have a common ontology, *i.e*., the same understanding of the data to be exchanged between the services. In our project, we are using the NATO Command and Control Information Interchange Data Model (C2IEDM) as the basis for an identifying tagset because we believe this standard offers the most likely path to a future common ontology for military simulations. The standard provides a way to extend the C2IEDM, so the ontology can grow as needed. Examples combining XML and C2IEDM are given in [10].

Our use of the C2IEDM applies the concepts of the Extensible Battle Management Language (XBML) described in [11]. In general, Battle Management Language (BML) is an unambiguous language used to

command and control forces and equipment conducting military operations and provide for situational awareness and a shared, common operational picture. BML is being developed as a standard representation of a "digitized commander's intent" to be used for real troops, for simulated troops, and for future robotic forces. BML is particularly relevant in a network centric environment for enabling mutual understanding. The XBML project has migrated BML into the XMSF domain. Three views are necessary to describe XBML completely:

Doctrine View – XBML must be aligned to doctrine. The vocabulary used to unambiguously generate executable tasks at the end of the process must be well defined in the context of the respective application domain.

Representation View – BML must model these aspects in a way that underlying Information Technology systems (M&S as well as Command and Control Information Systems) can exchange information and can make sense of the results. Therefore, BML uses an extended version of the Command and Control Information Exchange Data Model (C2IEDM) as the means for modeling.

Protocol View – BML must specify the underlying protocols for transferring BML information. The Extensible M&S Framework (XMSF) initiative has shown how BML representations can be transferred using Web-based, open standards.

In other words: XBML enhances C2IEDM from a semantic layer to an ontological layer, as proposed by Pohl in [12], while in the same time bridging the gap between command and control and simulation systems. This will enable XC2I to be used in an environment that includes simulations with tactical links for embedded training and decision support.

4. WSIM Architecture

The central purpose of XC2I is to implement Viewer/Controller Service that can be used to observe and control multiple Federates, all connected in a service-oriented architecture. Figure 2 shows our conception of the top-level architecture using WSIM. It has a WSIM Web service associated with each HLA federate, a WSIM client associated with each viewer, and a separate global Object ID server and Access ID server.

Figure 3 shows some details of the WSIM client and server architecture. The descriptions of the elements in the architecture are as follows.

The *Object ID server* coordinates object identities globally. It provides unique instance values with any object type, provides compressed identifiers for use with XML tag compression [13], and defines mapping of multicast groups to interest areas (see below). It also provides aggregation hierarchies (unit order of battle, in military parlance).

The *Access ID server* maintains access privilege information for the entire tagset by object type and issues access tokens supporting valid client information requests.

The *Access Control* element of the data server uses the access tokens to validate information requests before responses are passed through the WSIM server. Its client provides authentication information to be used by the server.

The *Area of Interest Management* element of the data server provides a filtering function, based on user information requests. Its client provides user information requests in the form of an Interest Management Language that we have defined, using the layered concept shown in Figure 4. This language is based on C2IEDM tags and is called C2 Interest Management Language (C2IML). The purpose of C2IML, like many XML-based languages, is to be a neutral translation mechanism between two systems with disparate interfaces, in this case the user's interest management language or interface, and the native interest management protocol. (The user may specify interest expressions via a graphical interface, such as the XC2I GUI, rather than in a written language.) Although XC2I is designed to support connection of individual viewers to a Web services enabled federation, C2IML is a general purpose interest management language for C2 and could be used to describe interest expressions between federates. Figure 4 shows both generic and JUO-specific mapping layers. The same generic mapping layers could also be used to implement C2IML in another federation.

The *Aggregation Interest Management* element of the data server aggregates low-level objects into composite objects, based on aggregation hierarchies. Its client provides user aggregation level requests.

The *RTI Interface* and *Generic Interface* provide for standard XML tagging of data that arrives with some other identification, such as HLA Federate Object Model (FOM) tags or command and control (C2) system data.

5. Aggregation Interest Management

AGIM is a novel aspect of our WSIM architecture. The M&S Glossary (DoDD 5000.59-M) [14] within the *US DoD Modeling and Simulation Master Plan* defines aggregation as "the ability to group entities while preserving the effects of entity behavior and interaction while grouped." Generally, aggregation is the composition of several lower level elements into a higher aggregate. An example is to aggregate the weapon systems into units; another is to aggregate lower level units, such as platoons, into higher level units, such as companies. Within XC2I we distinguish between type driven aggregation, based on the organization types of the simulated platform and entities established in the military order of battle, and instance driven aggregation, based on

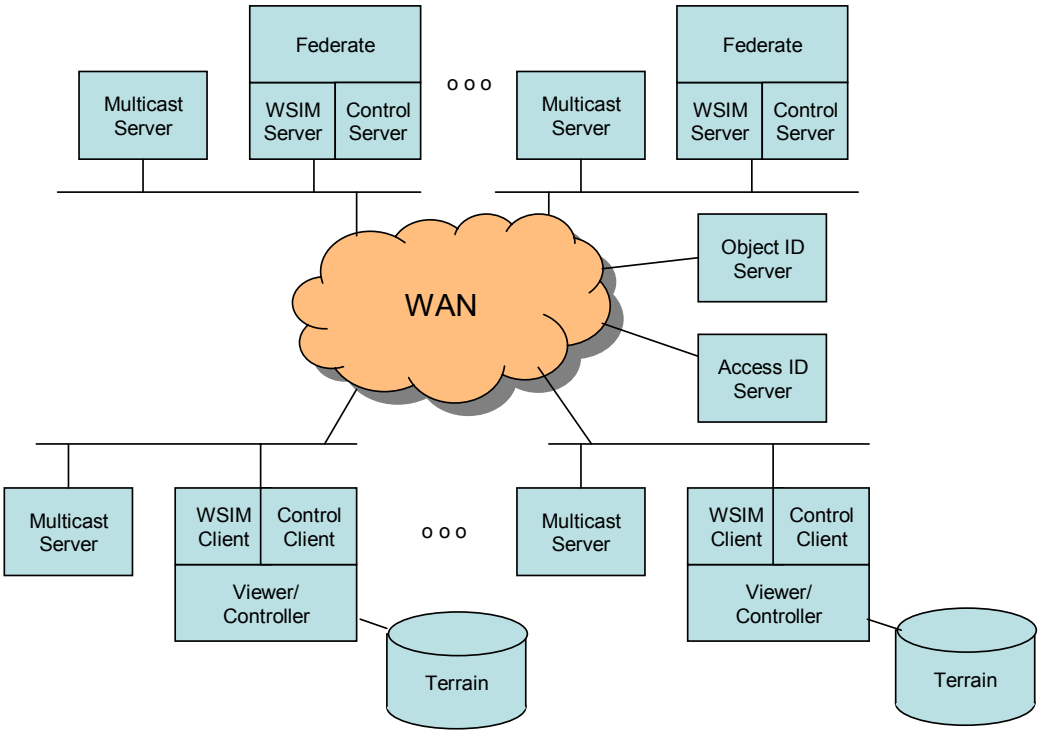

Figure 2. Top-level Architecture Using WSIM

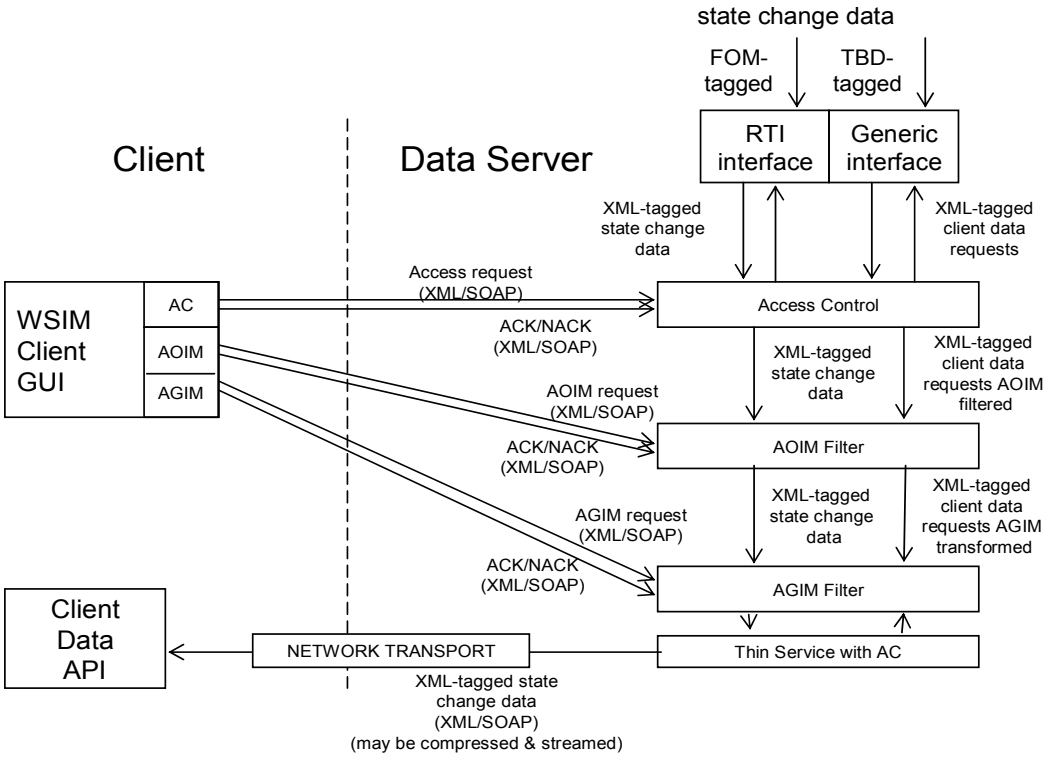

Figure 3. Detailed WSIM Architecture

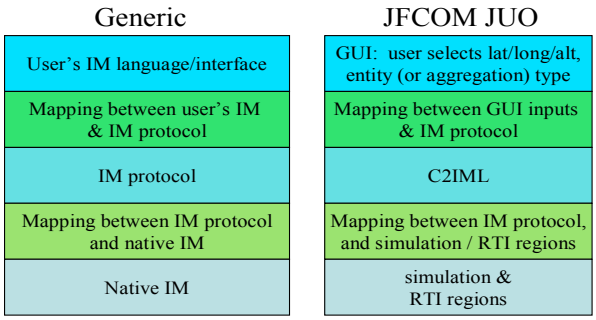

Figure 4. Generic IM Layers and Mapping Layers Specific to JUO

specific user requirements. Instance driven aggregation overrides type driven aggregation. Thus, a user may decide to display sister units as aggregates while the objects under his direct command are displayed in detail. Modern military operations, such as operations other than war or joint urban operations, further demand dynamic definition of new aggregates based on simulated (or observable) entities. An example could be crowds of non-combatants that are of interest to a local commander, but have no associated order of battle counterpart. Hybrids of the aggregates also are possible; a commander may be interested in the positions of his companies as well as of the tanks within those companies in the same display.

It is noteworthy that these WSIM principles are applicable, beyond simulations, to command and control interfaces generating a common operating picture based on a variety of resources, regardless whether a simulated or a real operation is monitored. The approach described here is applicable for dual use, both in distributed simulation and in distributed operations.

6. Role Based Access Control

The chief goals in defining the role based access control architecture for the authentication and authorization subsystem are:

- Defining a globally available identity management system;
- Associating user identities with simulation-specific roles; and
- Transparently limiting user interaction with the client based upon the user's available roles.

As an example, a user who is authorized to only view and control blue forces should not be presented with related options reserved for the red or white forces.

When a user is authenticated by the access control system, the user begins by entering a username and password at the client login prompt. This information is used to access a globally unique signed certificate that the user must implicitly present to the system, either via a secure keystore on disk or a physical token such as a smartcard. This certificate is sent to the access control server and used for identification services. As these certificates are already provided to many DoD personnel as physical tokens and similar authentication systems, and the certificate authority infrastructure is already in place for the DoD, the validity of the certificate can be checked by ensuring that an accepted authority signed it. The identification information found within the certificate can then be used to look up the user's privileges on the access control server for the requested system. While currently this lookup uses a relational database, other, more distributed identity management systems, such as LDAP, also can be used. Once the user has been authenticated and the available roles determined, the access control server returns a list of roles to the client, where each role is associated with an encrypted, signed token.

Once the list of available roles is received, the client offers the user an opportunity to select the preferred role. When the user selects a role, the client connects to the simulation via a top level Web Services Interest Management (WSIM) server, requesting that a session be started for the user with the given role. The request includes the token received earlier from the access control server. The WSIM server then verifies the authenticity of the token by verifying the validity of the signature over the token using the access control server's public key. If the access control server's key is not known, the WSIM server needs to contact the access control server directly to retrieve it for verification. The token is then sent on to the access control server to be unencrypted and verified. If the token is valid, the WSIM server caches the role information in the session information associated with the requesting client and connects to the simulation. The traditional exceptions that can be raised by an authentication system can be incorporated.

7. Streaming Delivery and WSIM

We have studied the relative efficiency of basic Web services compared to light-weight, streaming updates. Figure 5 shows the relative network traffic generated by the two approaches for a projectile trajectory, and clearly indicates that, where network resources are taxed by the volume of transactions, the streaming approach is superior.

We deal with the issue of network traffic in large-scale distributed simulations by introducing the concept of a Web service that exists to initiate a stream. There are two possible ways to approach streaming. Reference [15] reports on use of a Web service with the BEEP protocol to extend an HLA RTI via a Web service. A more aggressive approach, based on multicasting, is shown in Figure 6. Our WSIM architecture includes an option for delivering data as a multicast stream using compressed

Pure Web service		Web service plus streaming multicast	
Connect	136 bytes	Connect	136 bytes
HTTP Request Seg 1	1500 bytes	HTTP Request Seg 1	1500 bytes
Client Ack 1	40 bytes	Client Ack 1	40 bytes
HTTP Request Seg 2	120 bytes	HTTP Request Seg 2	175 bytes
Client Ack 2	40 bytes	Client Ack 2	40 bytes
HTTP Response Seg 1	833 bytes	HTTP Response Seg 1	835 bytes
HTTP Response Seg 2	40 bytes	HTTP Response Seg 2	40 bytes
Client Ack for seg 1	40 bytes	Client Ack for seg 1	40 bytes
Client Ack for seg 2	40 bytes	Client Ack for seg 2	40 bytes
Response 1	40 bytes	Response	40 bytes
Ack 1	48 bytes		
Response 2	48 bytes	Total for setup: 2886 bytes	
Ack 2	40 bytes		
		Multicast Packet Size average: 88 bytes	
Total Per Computation :	2829 bytes		
Grand Total	= 136 +350 * 2829	**Grand Total**	= 2886 + 350 * 88
	= 990286 bytes		**= 33686 bytes**

Figure 5. Network Traffic Generated by Pure Web Services Versus Streaming for a 350-point Projectile Trajectory

tags. Given that tens of XC2I viewers may be monitoring 10,000 or more objects each, over a wide area network as shown in Figure 1, multicasting is likely to be essential in order to support the user's requirements over the available network. We envision using the Overlay Multicast approach introduced in [16] to support WAN multicast data produced under the architecture of Figure 6.

8. Conclusions

Application of Web technologies to distributed simulation is still in its infancy. We have reported here on an approach that uses Web services, extended to support streaming multicast, as basis for sophisticated Interest Management in real time distributed simulation. We have implemented a partial prototype of this architecture, with promising results. Based on continued progress of this sort, we anticipate that Web-based modeling and simulation will grow into the full promise of composable distributed simulation, which will benefit from our work. We plan to report on our prototype in a future paper.

We anticipate that some of our architectural concepts will need to expand based on the realities of implementing the WSIM architecture. For example, the expression of C2IML is in the very early stages. It is limited to a small set of requests and we have not considered fully the implications of changing interest expressions or delivery of data that is out of scope for the viewer. These issues have been addressed in the HLA DDM services and lessons learned there need to be carried forward. Also the application of Overlay Multicast to meet the many-to-many networking requirement of real-time distributed simulation is a new approach that is likely to evolve over time. We look forward to making these concepts into reality as XMSF is applied to real-world problems such as the XC2I.

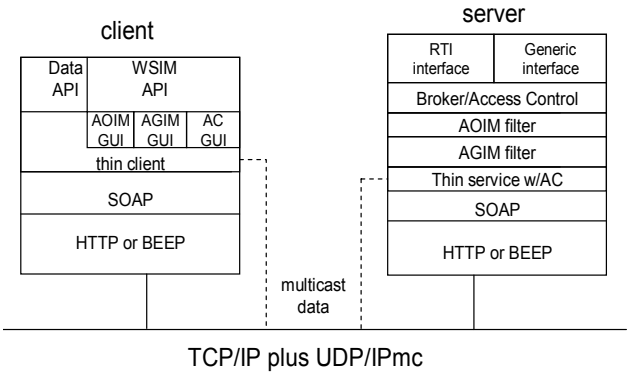

Figure 6. WSIM with Streaming Multicast

Acknowledgement

The work was supported in part by the US Joint Forces Command.

References

[1] D. Brutzman, M. Zyda, M., J. Pullen, and K. Morse, "Extensible Modeling and Simulation Framework (XMSF): Challenges for Web-Based Modeling and Simulation," US Naval Postgraduate School, 2002

[2] W3C (2002) Web Services Architecture Requirements, November 14 2002, online document at http://www.w3.org/TR/wsa-reqs.

[3] J. Pullen, *et. al.*, Using Web Services to Integrate Heterogeneous Simulations in a Grid Environment, *Proceedings of International Conference on Computer Science 2004*, Krakow, Poland, June 2004

[4] S. Singhal and M. Zyda, *Networked Virtual Environments: Design and Implementation*, ACM Press, 1999

[5] K. Morse, L. Bic and M. Dillencourt, "Interest Management in Large Scale Virtual Environments," *MIT PRESENCE - Teleoperators and Virtual Environments*, February 2000.

[6] Defense Modeling and Simulation Office, *US Department of Defense High Level Architecture Interface Specification*, 1998

[7] Institute of Electrical and Electronics Engineers, *IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – Federate Interface Specification*, IEEE Standard 1516.1, Piscataway, NJ, 2000

[8] K. Morse and J Steinman, Data Distribution Management in the HLA: Multidimensional Regions and Physically Correct Filtering, *Proceedings of the 1997 Spring Simulation Interoperability Workshop*

[9] IBM Corporation, Web Services Notification, http://www106.ibm.com/developerworks/library/specification/ws-notification/

[10] A. Tolk, XML Mediation Services utilizing Model Based Data Management, *Proceedings of the SCS Winter Simulation Conference*, Arlington, VA, December 2004

[11] M. Hieb, A. Tolk, W. Sudnikovich and J. Pullen, Developing Battle Management Language into a Web Service, *Proceedings of the 2004 Spring Simulation Interoperability Workshop*, Washington, D.C.

[12] J. Pohl, C2IEDM as a Component of the GIG Architecture, C2IEDM Workshop, Fairfax, VA, March 2004

[13] Open Source Development Network, XSBC: XML Schema-Based Compression, online document at https://sourceforge.net/docman/display_doc.php?docid=22640&group_id=86243

[14] US Department of Defense, M&S Glossary (DoDD 5000.59-M), *US DoD M&S Master Plan*, October 1995

[15] K**.** Morse, R. Brunton and D. Drake: Web Enabling an RTI – an XMSF Profile, Paper 03E-SIW-046, *Proceedings of the 2003 European Simulation Interoperability Workshop*

[16] D. Moen and J. Pullen, Enabling Real-Time Distributed Virtual Simulation over the Internet Using Host-based Overlay Multicast, *Proceedings of the 7th IEEE Distributed Simulation and Real Time Applications Workshop,* Delft, Netherlands, October 2003