CEWES MSRC/PET TR/98-44

Object Web (Java/CORBA) based RTI to support Metacomputing M&S

by

G. C. Fox
W. Furmanski
H. T. Ozdemir

Work funded wholly or in part by the DoD High Performance Computing Modernization Program CEWES Major Shared Resource Center through

Programming Environment and Training (PET)

Supported by Contract Number: DAHC 94-96-C0002
Nichols Research

Object Web (Java/CORBA) based RTI to support Metacomputing M&S

G.C. Fox, W. Furmanski and H. T. Ozdemir
Northeast Parallel Architectures Center, Syracuse University, Syracuse NY 13244-4100
gcf@npac.syr.edu, furm@npac.syr.edu

Abstract

We present here our Pragmatic Object Web based approach to High Performance Modeling and Simulation and we describe the associated middleware software recently prototyped at NPAC: JWORB (Java Web Object Request Broker) which integrates HTTP and IIOP protocols, and Object Web RTI which implements DMSO RTI 1.3 on top of the JWORB based CORBA / Java software bus. We explain how JWORB and OW RTI are used to build WebHLA – an interactive FMS training environment and we outline our plan towards WebHLA based Virtual Prototyping Environments for Testing, Evaluation and Simulation Based Acquisition.

Introduction

Virtual Prototyping Environments for T&E require integration of several M&S software modules, including simulators for tested objects, their parts, and suitable synthetic environments they operate in and interact with. Such integration of diverse simulation paradigms will be soon most conveniently performed within the new High Level Architecture (HLA) promoted by DMSO to enforce DoD-wide simulation interoperability. HLA includes the object models specified by the Object Model Template (OMT) and the Run-Time Infrastructure (RTI), acting as a software bus that supports interaction between HLA objects (federates) within applications (federations).

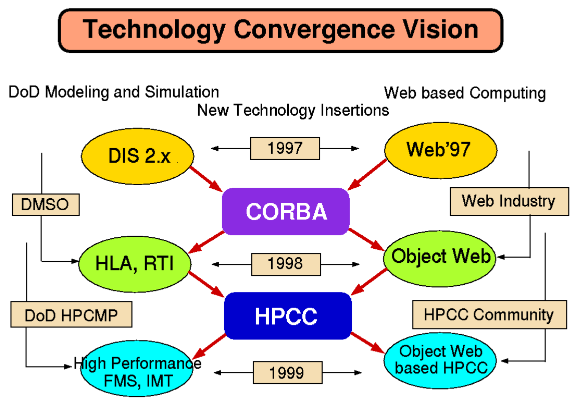

Fig 1: Technology Convergence Roadmap: Right and left columns illustrates the evolution of Web/Commodity and M&S technologies, respectively; middle column illustrates the major technology insertion thresholds that affect both evolution pathways.

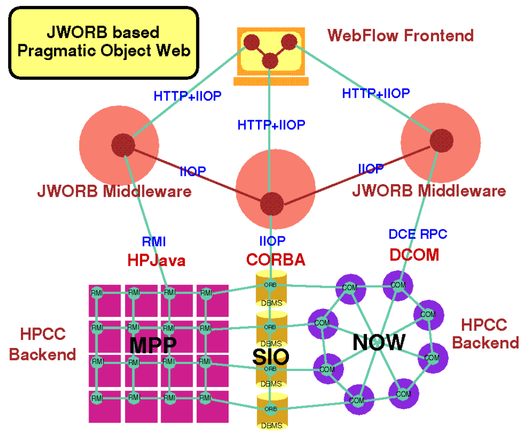

Fig 2: JWORB based Pragmatic Object Architecture: a mesh of JWORB servers connected via IIOP protocol forms the CORBA software bus in the middleware and links via HTTP+IIOP to the Browser front-ends and via the dedicated protocols to the legacy backends.

In parallel with these developments in the DoD M&S, new Web/Commodity standards are emerging or consolidating in the area of distributed objects and componentware such as CORBA, Java/RTI and DCOM (see Fig. 1). Current HLA is a custom distributed object model but DMSO's longer range plan includes transferring HLA to industry as CORBA Facility for Modeling and Simulation. Anticipating these developments, we are currently building as part of our HPCMP FMS PET activities at NPAC an Object Web based RTI prototype which builds on top of our new JWORB (Java Web Object Request Broker) middleware / integration technology.

JWORB is a multi-protocol Java network server, currently integrating HTTP (Web) and IIOP (CORBA) and hence acting both as a Web server and a CORBA broker. Such multi-server architecture enforces software economy and allows us to efficiently prototype new interactive Web standards such as XML, DOM or RDF in terms of a solid software engineering services and facilities of CORBA.

We are now testing this concept and extending JWORB functionality by building DMSO RTI support as a JWORB service. Such Object Web RTI uses Common Object Services of CORBA such as Naming, Event, Relationship, Notification etc. to build the core RTI objects such as RTIAmbassador, FederateAmbassador, RTIExecutive and FederationExecutive, using the Web/Java implementation framework.

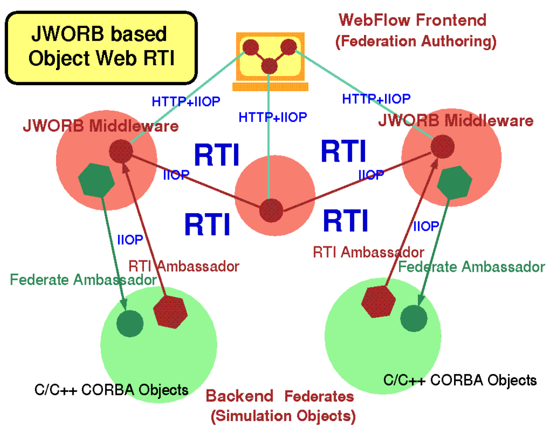

Fig 3: JWORB based Object Web RTI Architecture: RTI is implemented in Java and packaged as two major remote CORBA objects: Federate Ambassador and RTI Ambassador. Web front-ends such as WebFlow facilitateremote control, interactive steering and visual authoring of HLA applications

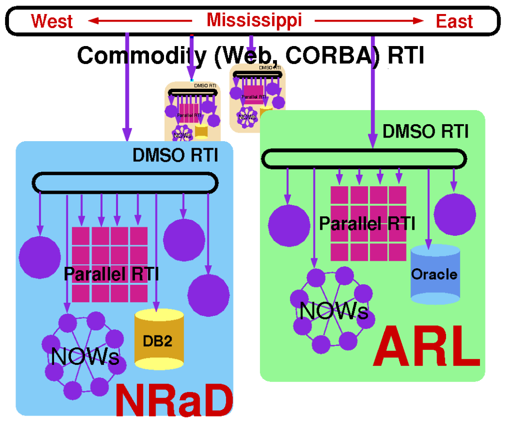

Fig 4: Object Web RTI based architecture for metacomputing FMS environment: M&S environments in the individual DoD labs, locally connected via DMSO RTI or HPC RTI, are interconnected via Object Web RTI to form a distributed metacomputer.

Due to the Web/CORBA integration enabled by JWORB, we will be able to inherit the growing arsenal of the Web/Commodity front-end technologies such as Java2D, Java3D or VRML. We are also working on CORBA/COM bridge support in JWORB which will enable interfaces to Microsoft graphics and multimedia technologies such as DirectX. Such a broad spectrum of commodity visualization techniques, integrated with the simulation engine/bus of RTI is of particular relevance for the T&E simulations that require a variety of front-end technologies to support object authoring, scenario development, runtime monitors and the analysis of the test databases.

Other JWORB related activities at NPAC include: a) new version of WebFlow, a visual dataflow authoring environment for 3-tier Web/Commodity based distributed computing, which uses the JWORB based middleware; and b) JWORB interface/wrappers for Globus Metacomputing Toolkit by Argonne, used in several HPC Labs [10]. Hence, our JWORB based RTI will offer a

natural integration platform for bringing together the HLA and HPC domains in service for advanced T&E applications such as Virtual Prototyping or Simulation Based Acquisition.
This paper presents the overview of NPAC JWORB activities, followed by the description of the early prototype of JWORB based Object Web RTI and our planned next steps.

1. Pragmatic Object Web

Recent developments in Internet/Intranet technologies start influencing the whole field of distributed computing, both in its enterprise and science & engineering domains. Most notably, Java appeared during the last few years as the leading language candidate for distributed systems engineering due to its elegant integrated support for networking, multithreading and portable graphical user interfaces.

While the "Java Platform" or "100% Pure Java" philosophy is being advocated by Sun Microsystems, industry consortium led by the OMG pursues a multi-language approach built around the CORBA model. It has been recently observed that Java and CORBA form a perfect match as two complementary enabling technologies for distributed system engineering. In such a hybrid approach, referred to as Object Web [1], CORBA is offering the base language-independent model for distributed objects and Java offers a language-specific implementation engine for the CORBA brokers, clients and servers.

Meanwhile, other total solution candidates for distributed objects/components are emerging such as DCOM by Microsoft or WOM (Web Object Model) by the World-Wide Web Consortium. However, standards in this area and interoperability patterns between various approaches are still in the early formation stage. For example, recent OMG/DARPA workshop on compositional software architectures [2] illustrated very well both the growing momentum and the multitude of options and the uncertainty of the overall direction in the field. A closer inspection of the distributed object/component standard candidates indicates that, while each of the approaches claims to offer the complete solution, each of them in fact excels only in specific selected aspects of the required master framework. Indeed, it seems that WOM is the easiest, DCOM the fastest, pure Java the most elegant and CORBA the most complete solution.

In our Pragmatic Object Web [3] approach at NPAC we adopt the integrative methodology i.e. we setup a multiple-standards based framework in which the best assets of various approaches accumulate and cooperate rather than competing. We start the design from the middleware which offers a core or a `bus' of modern 3-tier systems and we adopt Java as the most efficient implementation language for the complex control required by the multi-server middleware. We adopt CORBA as the base distributed object model at the Intranet level, and the (evolving) Web as the world-wide distributed (object) model. System scalability requires fuzzy, transparent boundaries between Intranet and Internet domains which therefore translates into the request of integrating the CORBA and Web technologies. We implement it by building a Java server (JWORB) [4] which handles multiple network protocols and includes support both for HTTP and IIOP. On top of such Pragmatic Object Web software bus, we implement specific computation and collaboratory services.

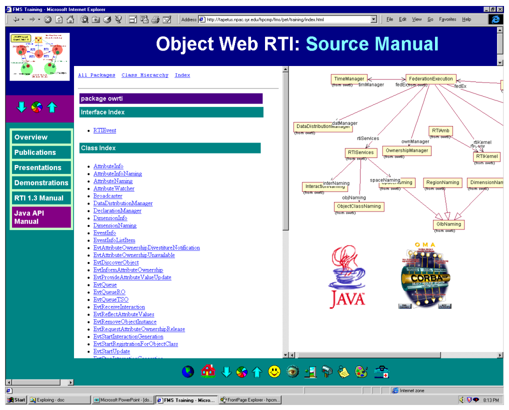

Fig 5: On-line software documentation of Object Web RTI within the WebHLA based FMS Training Space under development.

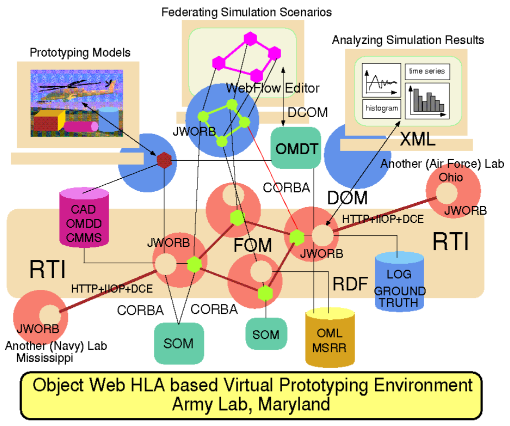

Fig 6: WebHLA based Virtual Prototyping Environment: a mesh of JWORB servers implements Object Web RTI in the middleware and links to a suite of Web based authoring, planning and analysis front-end tools, and to CORBA, Java COM or WOM wrapped legacy simulation backends.

2. JWORB (Java Web Object Request Broker) based middleware

JWORB [4] is a multi-protocol extensible server written in Java. The base server, illustrated in Fig. 2, has HTTP and IIOP protocol support. It can serve documents as an HTTP Server and it handles the IIOP connections as an Object Request Broker. As an HTTP server, JWORB supports base Web page services, Servlet (Java Servlet API) and CGI 1.1 mechanisms. In its CORBA capacity, JWORB is currently offering the base remote method invocation services via CDR based IIOP and we are now implementing the Interface Repository, Portable Object Adapter and selected Common Object Services.

After the core JWORB server starts up, it looks at configuration file to find out which protocols are supported and it loads the necessary protocol classes for each protocol (Definition, Tester, Mediator, Configuration). Definition Interface provides the necessary Tester, Configuration and Mediator objects. Tester object looks at the current connection's stream and decides whether it can interpret this connection or not. Configuration object is responsible for the configuration parameters of a particular protocol. Mediator object serves the connection. New protocols can be added simply by implementing the four classes described above and by registering a new protocol with the JWORB server.

After JWORB accepts a connection, it asks each protocol handler object whether it can recognize this protocol or not. If JWORB finds a handler which claims that it can serve this connection, then this protocol handler deals with this connection. Current algorithm looks at each protocol according to their order in the configuration file. This process can be optimized with randomized or prediction based algorithm. At present, only HTTP and IIOP messaging is supported and the current protocol is simply detected based on the magic anchor string value (GIOP for IIOP and POST, GET, HEAD etc. for HTTP).

3. High Level Architecture and Run-Time Infrastructure

High Level Architecture (HLA) [6] under development by the Defense Modeling and Simulation Office (DMSO) offers a common integration and interoperability platform for a broad spectrum of simulation paradigms. These include real-time (DIS) models used for combat training simulations, logical-time / event-driven models used for forces simulation, and faster-than-real-time models used in analysis simulations.

HLA is a distributed object technology with the object model defined by the Object Model Template (OMT) specification and including the Federation Object Model (FOM) and the Simulation Object Model (SOM) components. HLA FOM objects interact by exchanging HLA interaction objects via the common Run-Time Infrastructure (RTI) acting as a software bus similar to CORBA. Current HLA/RTI follows a custom object specification but DMSO's longer term plans include transferring HLA to industry via OMG CORBA Facility for Interactive Modeling and Simulation.

At NPAC, we are anticipating these developments are we are building a prototype RTI implementation in terms of Java/CORBA objects using the JWORB middleware. Although coming from the DoD computing domain, RTI follows generic design patterns and is applicable to a much broader range of distributed applications, including modeling and simulation but also collaboration, on-line gaming or visual authoring. From the HPCC perspective, RTI can be viewed as a high level object based extension of the low level messaging libraries such as PVM or MPI. Since it supports shared objects management and publish/subscribe based multicast channels, RTI can also be viewed as an advanced collaboratory framework, capable of handling both the multi-user and the multi-agent/multi-module distributed systems [7][8][9]. In the following, we summarize the ongoing NPAC work on JWORB based RTI prototype implementation.

4. JWORB based RTI Prototype at NPAC

RTI is given by some 150 communication and/or utility calls, packaged as 6 main management services: Federation Management, Object Management, Declaration Managmeent, Ownership Management, Time Management, Data Distribution Management, and one general purpose utility service.

Our RTI design, illustrated in Fig. 3, is based on 9 CORBA interfaces, including 6 Managers, 2 Ambassadors and RTIKernel. Since each Manager is mapped to an independent CORBA object, we can easily provide minimal support for distributed management by simply placing individual managers on different hosts.

The communication between simulation objects and the RTI bus is done through the RTIambassador interface. The communication between RTI bus and the simulation objects is done by their FederateAmbassador interfaces. Simulation developer writes/extends FederateAmbassador objects and uses RTIambassador object obtained from the RTI bus.
RTIKernel object knows handles of all manager objects and it creates RTIambassador object upon the federate request. Simulation obtains the RTIambassador object from the RTIKernel and from now on all interactions with the RTI bus are handled through the RTIambassador object. RTI bus calls back (asynchronously) the FederateAmbassador object provided by the simulation and the federate receives this way the interactions/attribute updates coming from the RTI bus.

Federation Manager object is responsible for the life cycle of the Federation Execution. Each execution creates a different FederationExecutive and this object keeps track of all federates that joined this Federation.

Object Manager is responsible for creating and registering objects/interactions related to simulation. Federates register the simulated object instances with the Object Manager. Whenever a new registration/destroy occurs, the corresponding event is broadcast to all federates in this federation execution.

Declaration Manager is responsible for the subscribe/publish services for each object and its attributes. For each object class, a special object class record is defined which keeps track of all the instances of this class created by federates in this federation execution. This object also keeps a seperate broadcasting queue for each attribute of the target object so that each federate can selectively subscribe, publish and update suitable subsets of the object attributes.
Each attribute is currently owned by only one federate who is authorized for updating this attribute value. All such value changes are reflected via RTI in all other federates. Ownership Management offers services for transfering, maintaining and querying the attribute ownership information.

We are currently implementing the Time Management service which offers support for logical time handling, and the data Distribution Management which offers advanced publishe/subscribe services via routing spaces or multi-dimensional regions in the attribute value space.

In parallel with the first pass prototoype implementation, we are also addressing the issues of more organized software engineering in terms of Common CORBA Services. For example, we intend to use the CORBA Naming Service to provide uniform mapping between the HLA object names and handles, and we plan to use CORBA Event and Notification Services to support all RTI broadcast/multicast mechanisms. This approach will assure quality of service, scalability and fault-tolerance in the RTI domain by simply inheriting and reusing these features, already present in the CORBA model.

5. Next Steps

We outlined here our Pragmatic Object Web based integration approach and we discussed in more detail the NPAC implementation of the RTI software bus in the JWORB framework. We intend to use our RTI both for DoD-specific high performance M&S applications within our WebHLA [8][9] project with the DoD High Performance Computing Modernization Office, and for other interactive distributed Web/Commodity application domains such as synchronous and asynchronous collaboration or visual dataflow authoring environments.

Our first application of Object Web RTI under development is the interactive FMS Training Space [9][13] (see also Fig. 5). This is based on a simple observation that HLA/RTI admits both live and synthetic participants and hence it offers a natural support for interactive hands-on distance training with the Web browser front-ends and Object Web RTI based middleware software bus. Such training space can be useful for exposing several ‘televirtual’ simulation environments, and is particularily suitable for teaching HLA itself as well as the associated FMS technologies such as SPEEDES or IMPORT. We call such environment WebHLA and we are currently building the Web linked database support for on-line documentation as well as simple interactive demos such as 3D-extended version of DMSO Jager [11]. In general, WebHLA students, instructors, mentors and the target systems being taught are mapped on individual

WebHLA federates and suitably synchronized via the RTI Management Services. Our first courses in the FMS Training Space will cover base HLA/RTI, followed by more advanced modules such as SPEEDES, E-ModSAF and IMPORT.

Having tested Object Web RTI in a multi-player interactive training environment, we will apply it in the next step for building Virtual Prototyping Environments for T&E applications such as currently planned in the Virtual Proving Ground project. The OW RTI based connectivity between various DoD labs participating in metacomputing scale simulations is presented in Fig. 4, whereas Fig. 6 illustrates the overall architecture of such systems. Middleware is given by a mesh of JWORB servers, implementing distributed RTI and acting as containers of WebFlow modules, suitably mapped on CORBA components, Enterprise JavaBeans or ActiveX controls. Only one lab activities are exposed and linked with other labs, participating in a joint VPE session. Three user workstations are engaged in the individual steps of the virtual prototyping cycle. The design station (left) offers VRML or DirectX/Chrome [12] based 'televirtual' tools to develop (or fine-tune in the runtime) an engineering model for a federate. The simulation station (center) offers WebFlow visual dataflow tools [11] to setup (or customize in the runtime) a federation representing a particular test operation procedure. The analysis station reads the simulation output data from either a log storage in an off-line mode or in the real-time from the RTI, and it offers an active compound document interfaces (perhaps using XML [13]) including dynamic plots, charts, monitors, steering controls etc.

Such VPEs for Testing, Evaluation and Simulation based Acquisition are clearly raising highly non-trivial software engineering, interoperability and integration challenges but we believe they can be realistically built now if we adhere to the emergent family of open standards coming from the several domains of the government, Web, commodity and enterprise computing and accumulated in our Pragmatic Object Web based WebHLA framework.

6. References

1. Robert Orfali and Dan Harkey, Client/Server Programming with Java and CORBA , 2nd Edition, Wiley 1998.

2. Craig Thompson, OMG/DARPA Workshop on Compositional Software Architectures, Monterey, CA January 6-8 1998

3. G. C. Fox, W. Furmanski, H. T. Ozdemir and S. Pallickara, *Building Distributed Systems for the Pragmatic Object Web*, book in progress, Wiley '98.

4. G. C. Fox, W. Furmanski and H. T. Ozdemir, JWORB - Java Web Object Request Broker for Commodity Software based Visual Dataflow Metacomputing Programming Environment , submitted for the HPDC-7, Chicago, IL, July 28-31, 1998.

5. D. Bhatia, V. Burzevski, M. Camuseva, G. Fox, W. Furmanski and G. Premchandran, WebFlow - a visual programming paradigm for Web/Java based coarse grain distributed computing , June '97, in the special issue of *Concurrency: Practice and Experience* on Java for Scientific Computing.

6. High Level Architecture (HLA) by the Defence Modelling and Simulation Office (DMSO)

7. D. Dias, G. C. Fox, W. Furmanski, V. Mehra, B. Natarajan, H. T. Ozdemir, S. Pallickara, Z. Ozdemir, Exploring JSDA, CORBA and HLA based MuTech's for Scalable Televirtual (TVR) Environments , presented at the Workshop on OO and VRML, VRML98 Conference, Monterey, CA, Feb 16-19,1998.

8. D. Bernholdt, G. C. Fox, W. Furmanski, B. Natarajan, H. T. Ozdemir, Z. Odcikin Ozdemir and T. Pulikal, WebHLA - An Interactive Programming and Training Environment for High Performance Modeling and Simulation , *in Proceedings of the DoD HPC 98 Users Group Conference*, Rice University, Houston, TX, June 1-5 1998.

9. G.C.Fox, W. Furmanski, S. Nair, H. T. Ozdemir, Z. Odcikin Ozdemir and T. Pulikal, "WebHLA - An Interactive Programming and Training Environment for High Performance Distributed FMS", to appear *in Proceedings of the Simulation Interoperability Workshop SIW Fall 98*, Orlando, FL, September 14-18, 1998.

10. G. C. Fox, W. Furmanski, T. Haupt, E. Akarsu and H. T. Ozdemir, "HPcc as High Performance Commodity Computing on top of integrated Java, CORBA, COM and Web standards", in *Proceedings of Euro-Par '98*, Southampton, UK, 1-4 September 1998

11. G. C. Fox, W. Furmanski, B. Goveas, B. Natarajan and S. Shanbhag, "WebFlow based Visual Authoring Tools for HLA Applications", to appear *in Proceedings of the International Test and Evaluation Association (ITEA) Workshop on High Performance Computing for Test and Evaluation*, Aberdeen MD, July 13-16 1998.

12. G. C. Fox, W. Furmanski, Subhash Nair and Z. Odcikin Ozdemir, "Microsoft DirectPlay meets DMSO RTI for Virtual Prototyping in HPC T&E Environments", to appear *in Proceedings of the International Test and Evaluation Association (ITEA) Workshop on High Performance Computing for Test and Evaluation*, Aberdeen MD, July 13-16 1998.

13. G. C. Fox, W. Furmanski and T. Pulikal, "Evaluating New Transparent Persistence Commodity Models: JDBC, CORBA PPS and OLEDB for HPC T&E Databases", to appear *in Proceedings of the International Test and Evaluation Association (ITEA) Workshop on High Performance Computing for Test and Evaluation*, Aberdeen MD, July 13-16 1998.