An Efficient Time Management Scheme for Large-Scale Distributed Simulation Based on JXTA Peer-to-Peer Network*

Azzedine Boukerche, Ming Zhang, Hengheng Xie
Paradise Research Laboratory
School of Information Technology and Engineering (SITE)
University of Ottawa, 800 King Edwards St.
Ottawa, Ontario, K1N 6N5, Canada
{boukerch, mizhang, hxie072}@site.uottawa.ca

Abstract

As an emergence technology, P2P is spreading to distributed simulation area, and many distributed simulation frameworks have used P2P as the middleware to interconnect their existing single processor's simulators to form distributed environments for simulation execution. In terms of simulation time management, most existing tools use a middleware layer to implement and support time management in a heterogeneous networking environment, and therefore, it is generally not easy to migrate a single processor's simulation to multi-processors in these frameworks. In this paper, we present a P2P based distributed simulation time management based upon JXTA API and Service Oriented Architecture (SOA), and we focus our discussion on how we implement the time management as a JXTA peer and a JXTA group service. Our time management is actually a native distributed message passing management framework, and does not rely on any middleware layer. Furthermore, we evaluate the performance of our implementations using a local Linux cluster. This work will establish a solid foundation for the more advanced distributed simulation services that have been proposed in our project [1].

1. Introduction

With the rapid development of computer network techniques and novel concepts of distributed computing, many of the modeling and simulation researchers are now facing new challenges on how to effectively use these new techniques to solve complex simulation problems. In particular, the development of new distributed simulation frameworks and tools is attracting more and more attentions because any

modeling and simulation application relies on an effective and easy-to-use execution platform for obtaining meaningful simulation results. The methodologies used in distributed simulations can be traced back to several decades ago, however, the advances in distributed simulation techniques are still limited by the available distributed computing techniques and computer resources. Peer-to-Peer (P2P) based network technology is attracting more and more attentions due to its feasibility to build large-scale autonomous software framework to meet the requirement of modern distributed applications. The P2P technology, such as Sun's JXTA, has been widely used as the fundamental framework to support higher level of distributed software tools including distributed file sharing system, distributed network management system, as well as distributed simulation tools. Indeed, the P2P network computing technique and novel Service Oriented Architecture (SOA) open a new direction for creating more effective and flexible distributed simulation frameworks. For instance, the Discrete Event System Specification (DEVS) [2] based distributed simulation has exploited all these possibilities in the newly developed tools including DEVS/P2P [3], DEVS/GRID [4], DEVS/SOA [5], etc. Meanwhile, distributed simulations based on other methodologies, such as [6][7], are also applying these novel distributed computing concepts.

It is worth noting that the simulation time management is the key in any simulation tool, and it basically provides the minimum execution environment for simulation applications. Indeed, time managements is the most important component in a distributed simulation framework, and therefore, has been thoroughly studied by many researchers [8][9]. In this paper, we focus our discussion on the development of a conservative simulation time management in a highly dynamic JXTA based P2P network environment, and

* This work is partially supported by Canada Research Chair Program, NSERC, Ontario Distinguished Researcher Award, Ontario Early Researcher Award, EAR Award and ORF Funds.

we will present the details of our implementations with some basic performance evaluations.

The reminder of this paper is organized as follows: Section 2 introduces related background with a focus on JXTA and some recent distributed simulation tools in terms of time management; Section 3 presents our implementations of time management in a JXTA framework; Section 4 demonstrates our basic performance evaluation of the time management implemented as JXTA group service; Section 5 concludes this paper with suggestion for some future work.

2. Background

2.1 P2P Network and JXTA

Peer-to-Peer (P2P) based networking technology is becoming dominant in distributed computing area. Compared to traditional client/server based application architecture, P2P system is able to provide more flexible and scalable framework to support large-scale and complex distributed applications. As one of the representative P2P techniques, JXTA, an application framework developed by Sun [10,11], becomes an dominant tool for rapidly developing advanced P2P applications. Indeed, JXTA provides a set of protocols to support network peers that can discover, communicate, and monitor one with another. As shown on Figure 2-1, JXTA can provide “peer group” based virtual network, which can be used for designing highly dynamic and self-aware heterogeneous virtual computing infrastructure. Such a virtual distributed network aims to support run-time evolution of participating computing entities through publish/subscribe based peer discovery protocol, as well as *Pipe* based message passing mechanism.

It worth mentioning that JXTA uses XML format for common message encoding and service advertisement, which makes it capable of supporting interoperability, platform independency, and ubiquity [10]. JXTA has six built-in protocols to enable autonomous P2P networking, which includes: *Peer Discovery Protocol (PDP)*, *Peer Information Protocol (PIP)*, *Peer Resolver Protocol (PRP)*, *Pipe Binding Protocol (PBP)*, *Endpoint Routing Protocol (ERP)*, and *Rendezvous Protocol (RVP)* [11].

The evaluation of JXTA performance has attracted many researchers in order to further consider its feasibility of being used for high performance backbone for distributed applications. [12][13][14] benchmarked JXTA in different types of interconnected networks including Grid infrastructure.

Figure 2-1 JXTA Virtual Network [1]

It was found that JXTA was suitable for delivering high performance in a System Area Network (SAN) or Wide Area Network (WAN) after fine-tuned for its internal communication mechanism.

2.2 Time Management in Distributed Simulation Tools

Recent advances in distributed simulation involve using novel concepts of distributed computing combined with traditional methodologies of parallel and distributed simulations. Indeed, many of the recent distributed simulation tools use network middleware to bridge the single processor's simulators for message passing in a distributed network environment. For instance, [15] implemented HLA's key functionalities as Grid service, thus, provided a distributed simulation framework based on invocations of Grid services. Other tools, such as DEVS/SOA, DCD++[16], used the same strategy to wrap their existing single processor's simulators as standard web services to form an integrated distributed simulation environment with the Internet as backbone.

Based on such approaches, the time managements in these tools, in general, conform to the algorithms used in their corresponding single processor's simulators. For instance, the time management used in DEVS/P2P used the same conservative algorithm as used in single machine's DEVS, such as DEVSJAVA [17]. In contrast, HLA based distributed simulation tools used either conservative, optimistic or their combination in their

Figure 3-1 Pipe Based Configuration of Simulation Components in JXTA Network

time management algorithms. It worth noting that all these tools have a middleware layer to implement and support time management in a heterogeneous networking environment, and therefore, it is generally not easy to migrate a single processor's simulation to multi-processors in these frameworks. As will be discussed in the following sections, we use a different approach to implement the time management in a distributed environment, in which the simulation entities interact directly with the time management service without using any middleware. Such approach is able to provide more dynamic and flexible distributed simulation framework to support fault tolerance, run-time reconfiguration, etc.

3. Implementations of Time Management in JXTA Framework

In this section, we introduce and present our implementation of time management in a highly dynamic JXTA framework. Our approach is based on the concept of service components described in our proposed novel distributed simulation architecture [1]. As shown in Figure 3-1, any simulation entity or simulation service is designed and implemented as an individual component with its own unique input/output pipes for message passing. These pipes can be bound together either statically or dynamically. The concept of “plug-and-play” is used to support high level reusability and interoperability of simulation components. For instance, we can easily replace one time management implementation with another without affecting any other parts of the system as long as the replaced one conforms to the same input/output pipes’ definitions. Based on such design philosophy, we can implement the time management either as a standard entity or as a service. The difference is that the service based implementation provides a higher level of

abstraction and also is able to support autonomous service discovery. Furthermore, Figure 3-2 demonstrates our JXTA group based virtual simulation network, in which the simulation entities and Time Management Service (TMS) can reside on any computing node with no regard to the IP and port assignment. The TMS handles the time advance and synchronization of all involved simulation entities, while the message passing or event propagation happens directly among the entities. The simulation entities and TMS can also interact directly with other simulation services whenever necessary. As we can see, such approach makes the deployment and partition of simulation entities much easier than those middleware based solutions.

Basically, we implemented the time management both as simulation entity and as JXTA group service. We did not notice significant performance difference in terms of overall simulation execution time in a small cluster, and the detailed experiments will be discussed in the next section. As a matter of fact, we implemented the time management using a centralized conservative algorithm, in which the casualty is strictly prohibited. One of the advantage of this approach is that the fault tolerance can be easily achieved by using redundant TMS in the network. Indeed, the TMS in our implementation is a lightweight service component that is basically stateless. We present our time management algorithm in Table 3-1.

4. Experiment

We have presented our implementations of JXTA based time management in previous section. We hereby demonstrate our preliminary results obtained at a small scale Linux cluster. The Linux cluster has multiple computing nodes, each of which has an Intel Xeon 3.4G CPU with 2G memory. The installed

Figure 3-2 JXTA Group Based Time Management Service

Definition:

Time Management Service=TMS;

Time of Next Event= T_N ;

Minimum of Time of Next Event Among all Simulation Entities= $\min T_N$;

Steps to advance simulation time:

1. Each simulation entity send "registration" request to TMS;
2. if(number of registered entity == predefined number of entities in TMS)
TMS send T_N request to all simulation entities;
3. Each entity reply TMS with its T_N ;
4. TMS calculates $\min T_N$;
5. TMS sends $\min T_N$ to all simulation entities;
6. For each simulation entity

{

If ($T_N == \min T_N$)

Execute State Transition Function(also update T_N);

Execute Message Out Function;

Else

Send "done" message to TMS;

}

{

If (get *Event* message from other entities)

Execute Event Function;

Send "finished" message to other entities.

If (get "finished" message from all other entities)

Send "done" message to TMS;

}
- End For
7. if(TMS get "done" message from all entities)
TMS send T_N request to all simulation entities;
Loop back to step 3

Table 3-1 Time Management Algorithm

Total No. of Entities	Total No. of Nodes	Node 1	Node 2	Node 3	Node 4
5	1	TMS+5 entities			
5	3	TMS	1 “generator”	4 “processor”	
5	4	TMS	1 “generator”	2 “processor”	2 “processor”
11	1	TMS+11 entities			
11	3	TMS	1 “generator”	10 “processor”	
11	4	TMS	1 “generator”	5 “processor”	5 “processor”

Table 4-1 Test Configuration

Figure 4-1 Simulation Execution Time for Different Configuration

operating system is GNU/Linux 2.6.11. We use JXTA Java Library 2.5_RC1 and JDK1.6.0_04 to perform our test.

We first use two simulation entities, a “generator” and a “processor”, to test and validate our TMS in a local cluster environment. Both of the simulation entities are standard simulation entities that conform to our framework. The “generator” can periodically generate *task* message from its output pipe, while the “processor” accepts *task* message from its input pipe, then processes the *task* and generates output (optional) to its output pipe. Through simulating the two simulation entities in the cluster, we validate that our TMS works correctly.

We then use one “generator” and multiple “processor” to evaluate the performance of our time management service. The test configuration is shown in Table 4-1.

Figure 4-1 demonstrates the simulation results we obtained in a multi-nodes Linux cluster environment. We find that our TMS runs much faster when it resides on a separated computing node, and then communicates with involved simulation entities through JXTA network. We also compare the two implementations (as service and as non-service) of time managements and find that their performance are very close, which means that the overhead of service discovery in a small local LAN is small. As shown in

Figure 4-1, using more computation nodes can significantly reduce the overall simulation execution time in our JXTA-based simulation framework, and this phenomenon is consistent with increased number of simulation entities. Basically, the experiment result presented in this paper provides an initial performance evaluation of our P2P based distributed simulation tool.

5. Conclusion and Future Work

In this paper, we proposed and implemented a centralized conservative time management in a JXTA-based P2P network environment. Our design follows Service Oriented Architecture (SOA), and our implementation is based on Sun's JXTA API. We found that the JXTA platform could provide a very effective backbone for building complex simulation services, in particular, the distributed simulation time management. Our basic and initial performance evaluation demonstrated the potential of using JXTA to build scalable and flexible time management in a heterogeneous network environment.

For the future work, we will do a thorough performance evaluation using more complex simulation entities. We will also develop time management service using other parallel and distributed algorithms.

References

- [1]. A. Boukerche, M. Zhang, "Towards Peer-to-Peer Based Distributed Simulations on a Grid infrastructure", 41st Annual Simulation Symposium (ANSS-41 2008), pp. 212-219.
- [2]. B. P. Zeigler, T.G. Kim, and H. Praehofer, "Theory of Modeling and Simulation", 2 ed., New York, NY: Academic Press, 2000.
- [3]. S. Cheon, C. Seo, S. Park, B. P. Zeigler, "Design and Implementation of Distributed DEVS Simulation in a Peer to Peer Network System", 2004 Military, Government, and Aerospace Simulation.
- [4]. C. Seo, S. Park, B. Kim, S. Cheon, B. P. Zeigler, "Implementation of Distributed High-performance DEVS Simulation Framework in the Grid Computing Environment", 2004 High Performance Computing Symposium.
- [5]. S. Mittal, J. L. Risco-Martin, B.P. Zeigler, "DEVS-Based Simulation Web Services for Net-centric T&E", Summer Computer Simulation Conference SCSC'07, July 2007.
- [6]. K. Zajac, A. Tirado-Ramos, Z. Zhao, P. Sloot, M. Bubak, "Grid Services for HLA-Based Distributed Simulation Frameworks", LNCS 2970, pp. 147-154, 2004.
- [7]. K. Rycerz, M. Bubak, M. Malawski, P. Sloot, "A Framework for HLA-Based Interactive Simulations on the Grid", SIMULATION, Vol. 81, No. 1, 67-76 (2005).
- [8]. Y. Wang, S. J. Turner, W. Cai, X. Chen, "Time management in a service-oriented architecture for distributed simulation on the grid", Proceedings of the 2007 summer computer simulation conference, Pages 392-399.
- [9]. R. M. Fujimoto, "Time Management in The High Level Architecture", SIMULATION, Vol. 71, No. 6, 388-400 (1998).
- [10]. JXSE 2.5 Programmers Guide: JXTA Concepts, online at: https://jxta-guide.dev.java.net/source/browse/*checkout*/jxta-guide/trunk/src/guide_v2.5/JXSE_ProgGuide_v2.5.pdf
- [11]. JXTA Protocols Specification, online at <http://jxta-spec.dev.java.net>.
- [12]. E. Halepovic, R. Deters, "The costs of using JXTA", Proceedings of the Third International Conference on Peer-to-Peer Computing, 2003. (P2P 2003), 160 - 167.
- [13]. G. Antoniu, M. Jan, D. A. Noblet, "Enabling the P2P JXTA Platform for High-Performance Networking Grid Infrastructures", Lect. Notes in Comp. Science, September 2005, No. 3726, p. 429-440.
- [14]. G. Antoniu, P. Hatcher, M. Jan, D.A. Noblet, "Performance Evaluation of JXTA Communication Layers.", 5th International Workshop on Global and Peer-to-Peer Computing (GP2PC '05).
- [15]. K. Pan, S. J. Turner, W. Cai, Z. Li, "A Service Oriented HLA RTI on the Grid", 2007 IEEE International Conference on Web Services (ICWS 2007).
- [16]. J. Chazal, L. Quinet, Q. Liu, M. K. Traoré, G. Wainer, "Performance Analysis of Web-based Distributed Simulation in DCD++: A Case Study across the Atlantic Ocean", HPCS, part of Spring Simulation Multiconference 2008.
- [17]. B.P. Zeigler, H. S. Sarjoughian, "Introduction to DEVS Modeling & Simulation with JAVA", ACIMS publication, online at: http://www.acims.arizona.edu/SOFTWARE/devsjava_licensed/CBMSManuscript.zip.