Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/072L32

A Run-time Infrastructure based on Service-Distributed Architecture

Zhiteng Wang[1], *Hongjun Zhang*[1], *Rui Zhang*[1], *Yong Li* [2] *and Baoyu Xu* [1]

[1]PLA University of Science and Technology, Nanjing, China
[2]Nanjing Artillery Academy, Nanjing, China

Received: 4 Oct. 2012, Revised: 12 Jan. 2013, Accepted: 15 Jan. 2013
Published online: 1 Jun. 2013

Abstract: This paper points out the deficiencies of traditional run-time infrastructure (RTI) and Service-oriented High Level Architecture/Run Time Infrastructure (HLA/RTI) based on Web Services developed by researchers. The deficiencies of RTI involve load balancing, real-time interaction, safety and stability, and so on. This paper proposes Service-Distributed Run-time Infrastructure (SDRTI), which deploys RTI simulation services on the Internet, as a means of solving these problems. SDRTI has six key technologies: model mapping mechanisms, Simulation Service Management Bus (SSMB), traverse Network Address Translators (NATs) and Firewall's Firewall (FW) in public networks, encrypted communications, decoupled simulation service, and optimized network traffic measures. The response speed and loading ability of SDRTI was evaluated using the program Simple Collision in the enclosure space. The simulation results indicate that SDRTI is effective and feasible.

Keywords: HLA/ RTI, Web Service, Simulation service, Service-Distributed

1 Introduction

Distributed Interactive Simulation (DIS) adopts consistent structures, standards, protocols, and databases, and connects the simulation equipment with a local area network (LAN) or wide area network (WAN) to improve the interoperability and reusability of the dispersal simulation components in the simulation application, as well as to achieve a participatory synthetic simulation environment [1] [2][38][39]. The United States Department of Defense (DOD) published M&S Master Plan (MSMP) and established an universal simulation technology architecture to solve many interoperability problems of the internal distributed simulation application. The core of this plan is High Level Architecture (HLA). The HLA Run-Time Infrastructure (RTI) is a software implementation of the Interface Specification that provides services defined in the Interface Specification, including services to start and stop federation execution, services to send data between interoperating federates, services to control the amount and routing of data passed, and services to coordinate the passage of simulated time among the federates. Federates perform these functions by invoking the appropriate RTI service. The RTI may also invoke services provided by the federate, such as receiving data and services which are likewise defined in the Interface Specification. Different groups developed different kinds of RTIs, such as RTI 1.3 [3], MÄK RTI [4], pRTI1516 [6] [7], KD-RTI [8] [9], BH-RTI [10], and GY-RTI [11]. These RTIs have been used in a variety of live, constructive, and virtual simulations involving scientific analysis, experim-ental research, training, and other activities. In 2000, HLA was accepted as an Institute of Electrical and Electronics Engineers (IEEE) standard
(IEEE1516-2000) [5]. The structural model of RTI can be classified into three types: central, distributed, and hierarchical. Most RTIs adopt a central structure. However, their drawbacks have been revealed in practice and application for the following reasons [12]:

1. Because HLA defines the RTI interface as a special programming language binding and platform binding, RTI is dependent on the special programming language and platform.

2. The central mode of RTI becomes a bottleneck. Stored and processed data become problems for the central server, where the interactive data increase on a massive scale to maintain simulation granularity and accuracy. A breakdown in the central server causes the

* Corresponding author e-mail: wangzhiteng168@163.com

simulation application to crash. This load balancing deficiency leads to heavy loading in the central server when the simulation nodes increase.

3. Most RTIs use a static-linking mode that decreases the RTI portability of the federates.

4. Because the RTI only supports distributed applications based on the local area network or special network, the RTI cannot meet the challenges posed by a well-distributed simulation application, whether cross-specialty or cross-disciplinary.

2 Related research on service oriented HLA/RTI

Researchers have proposed an approach that combines the idea of Service Oriented Architecture (SOA) with HLA/RTI [13] to solve problems associated with traditional RTI. Many researchers have constructed Service Oriented HLA RTI (SOHR) by using Web Services [40]. Morse achieved simulation component communication with RTI over a WAN by constructing a web-enabled RTI based on the SOAP and BEEP web communication protocols [14]. Möller introduced the web services-based HLA Evolved application programming interface (API), and researched three methods of integrating HLA into a web-based service: (1) using Web Services alone, (2) using Web Services to bridge an HLA system and an external system, and (3) using the HLA Evolved Web Service API [15][16][17]. M. Dragoicea proposed the integration of HLA and SOA into a Simulation Framework [18]. W. Zhang addressed the problems of deployment mode, data coding, data exchange mode, and invocation state in the process of developing service-oriented HLA [19][20]. Z.J. Jiang proposed an extended RTI solution that allowed users to invoke web service RTI over the internet by modifying the RTI interface to act as a web service [21]. X. Zhou anticipated web services-based distributed RTI, which distributes many RTI services over multiple Internet servers [22]. S.C. Tang proposed an extended HLA multilayer federation integration architecture (MLFIA), which, combined with SOA and HLA, forms a four-layer collaborative simulation platform structure that effectively integrates the model's resources [23]. The integration of HLA and SOA was proposed by H.M. Zhang to achieve better interoperability and reusability among heterogeneous simulation components in a distributed environment [24] [25]. In 2008, the IEEE modified the HLA1516-2000 standard, updating it for the next-generation HLA Evolved standard. Many researchers have also realized the drawbacks of web services-based SOHR. Turner revealed the defects of web service communication protocols in engineer applications [26]. According to L.J. Xu, service-oriented HLA using web services are only appropriate for simulation applications that require coarse grain, low data update frequency, or non-real-time requests [36]. Byrne noted safety and stability drawbacks in simulation applications based on web services [27][31][32]. B.H. Jin pointed out load balance defects in web service-based simulation applications[28]. S.Q. Di suggested that the intrinsic characteristics of web services, such as being real-time and stateless, lead to problems in simulation models [37].

In fact, service-oriented services not only include web services, but generalized services as well. Packages of other components, such as EJB, JMS, JavaBean, COM/DCOM, CORBA, and ICE, are also categorized as services [29][33-35]. Services in the field of simulation and modeling are generalized services, which include web services. The relationship between simulation and SOA models is shown in Figure 1. This article proposes SDRTI, whose idea come from service-oriented architecture, to address many inadequacies of traditional RTI in large scale simulation, such as heterogeneous communication, and highly distributed simulation [26][27][28][30][31] [32][36] [37]. The main goal of this article is to achieve SDRTI, which has many excellent characteristics, such as heterogeneous communication, load balancing, loosely coupled services, reusability, high security and stability, wide area network, stable. and efficient network communication speed.

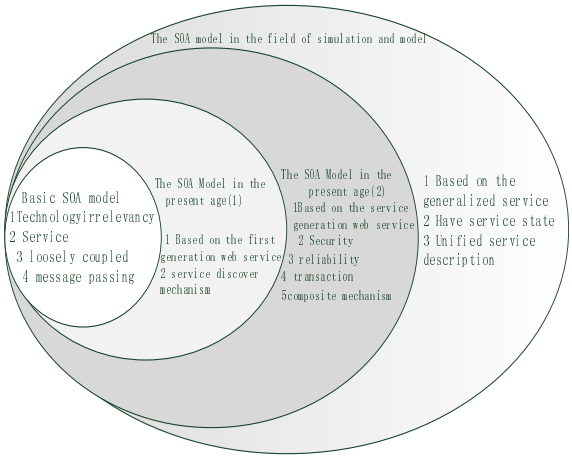

Fig 1. The SOA model in the field of simulation and model

3 Six key technologies for Service-Distributed Run Time Infrastructure

The main concept of the SDRTI is that services in RTI are decoupled to federation management serv-ice, time management service, object management service, ownership management service, interactive class management service, and so on and services are distributed in different nodes on the internet, as shown in Figure 2. The foremost merit of this method is that it overcomes the drawback of a central RTI and offer an effective load balancing solution where simulation

services are distributed among the different nodes on the Internet.

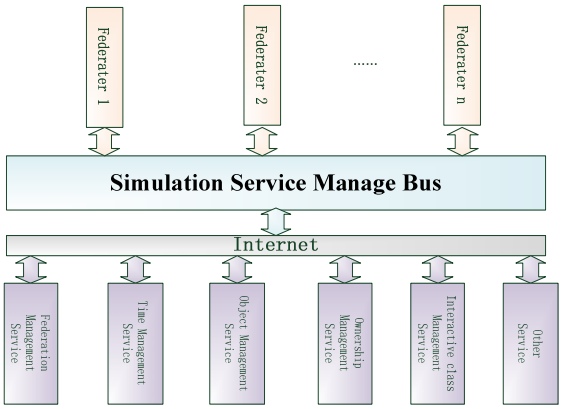

Fig. 2 The SDRTI framework SDRTI has six key technologies: a model mapping mechanism, SSMB, the ability to traverse firewalls in public networks, encrypted communication, decoupled services, and size-optimizing communication. Supported by the model mapping mechanism, SDRTI can achieve communication among different languages and platforms. SSMB is responsible for managing distributed simulation services by means of a service directory, state monitoring, and service scheduling functionality. The routefirewall service is responsible for achieving traversing the Firewall and Network Address Translator (FW/NAT). The Secure Socket Layer (SSL) protocol is applied to ensure communication safety. The data transfer service is responsible for decoupling the different simulation services. Choosing the most effective mode of message passing on the Internet decreases network traffic and increases the efficiency and stability of communication. Each key technology is discussed in detail as follows.

3.1 Model mapping mechanism

The model mapping mechanism is the fundamental abstraction mechanism for separating object inter-faces from their implementations. It establishes a contract between federates and the simulation that describes the types and object interfaces used by a specific application. This description is independent of the implementation language, so it does not matter whether the federate and simulation were written in the same language as the SSMB. The Interface Description Language (IDL) definitions are compiled for a particular implementation langu-age by a compiler. This compiler translates the language-independent definitions into language-specific type definitions and interface functions. These types and interface functions are used by the developer to provide application functionality. The translation algorithms for the various implementtation languages are known as language mappings. Currently, the model mapping model defines language mappings for C++, Java, C#, Python, and PHP. In SDRTI, after establishing the distributed object model of the RTI in accordance with RTI standards, the distributed object model describes the corresponding RTI service and the dependent specific data construction through the IDL definitions, and saves

the data in a description file. The interface of the RTI ambassador is described below:

interface RtiAmb

{void createFederationExecution(string executionName, string FED) throws RTIexception; .....};

Using model mapping mechanisms, the interface can be mapped into the corresponding language, such as C++, C#, Java, PHP, and Python, as shown in Figure 3. This mechanism enables cross-language coding mapping.

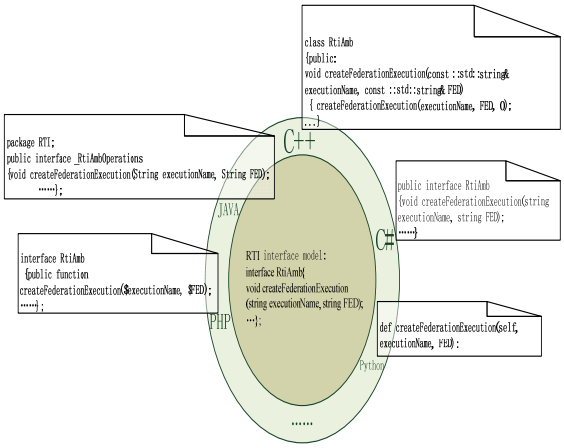

Fig.3 Model mapping mechanism schematic plan

According to HLA rules, federates cannot directly communicate through RTI. Therefore, achieving interoperability among different federates with different languages is actually making RTI interoperate with different federates with different languages. Taking C++ and Java as an example, assuming that the Central RTI Component (CRC) is implemented in the C++ language and the federate is implemented in Java, the relationship of the RTI interface language mapping is shown in Figure 4.

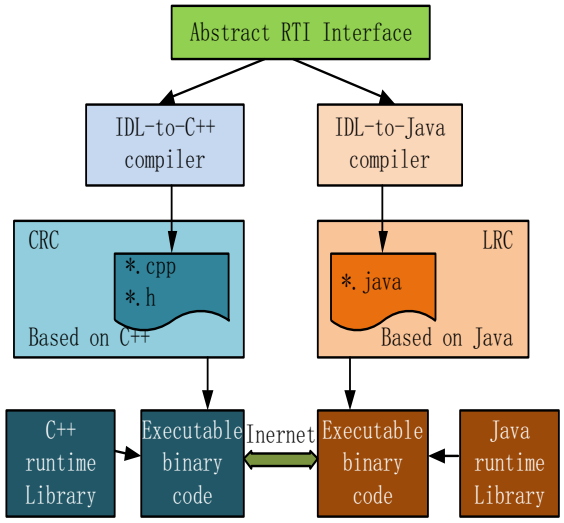

Fig.4 RTI interface mapping to different developer languages

All codes are converted to the unified code of the RTI core communication. Because all the units communicate with each other using this unified code, various units using different developer languages and platforms can communicate, as shown in Figure 5.

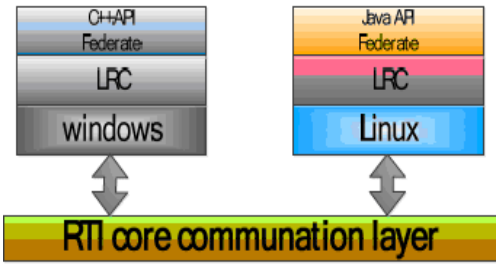

Fig.5 Cross-language, cross-platform communication schematic plan

3.2 Simulation Service Management Bus (SSMB)

The SSMB is an import part of the SDRTI, which can support the synchronous or asynchronous call of the simulation services through multiple protocols. The SSMB also manages the distributed simulation services. The core function of the SSMB includes the service register, service work record, service state monitor, service scheduling management, and service request listen, as shown in Figure 6. The simulation service registers at the SSMB using the standard service description language. The service register information contains detailed information such as name, address, port, identity, protocol, service description, and so on. When the federate applies the simulation service to the SSMB, the SSMB finds the service on the register list. Once the SSMB finds the simulation service, it establishes a connection for the federate based on the name, address, and port of the simulation service. The service work record records the efficiency of the simulation in achieving the work. This record contains the service stability, probability of achieving the work, and so on. The SSMB can comprehensively evaluate the efficiency of the simulation service via the work record, and the evaluation result is taken as important evidence for service scheduling management. The service state monitor monitors the simulation service in real time and reports the service state to the SSMB. This measure lowers the chance of SSMB errors and increases system stability. The service request listen listens for messages from federates requesting for a simulation service. The service scheduling management chooses the optimal simulation service for the federate by using an optimal algorithm.

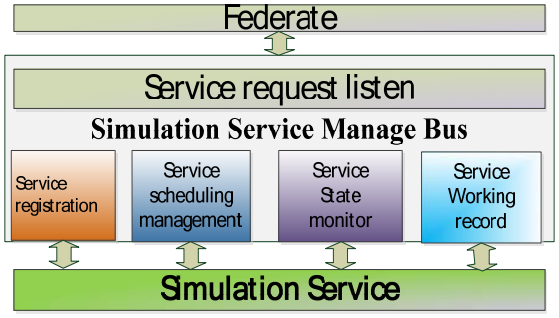

Fig.6 STRUCTURE OF THE SSMB

3.3 Communication security

Security is an important consideration in distributed simulation applications, both within corporate intranets and over untrusted networks such as the Internet. The ability to protect sensitive information, ensure its integrity, and verify the identities of the communicating parties is essential to developing secure applications. The Secure Socket Layer (SSL) protocol is the de facto standard for secure network communication. SSL's authentication, nonrepudiation, data integrity, and strong encryption functions make it the logical choice for securing SDRTI. When a federate or simulation establishes an SSL connection to the SSMB, a handshake is performed. During a typical handshake, digital certificates identifying the communicating parties are validated and symmetric keys are exchanged for encrypting the session traffic. Public key encryption, which is too slow to be used for the bulk of a session's data transfer, is used heavily during the handshaking phase. Once the handshake is complete, SSL uses message authentication codes to ensure data integrity, allowing the federate, simulation, and SSMB to communicate at will with reasonable assurance that their messages are secure, as shown as Figure 7.

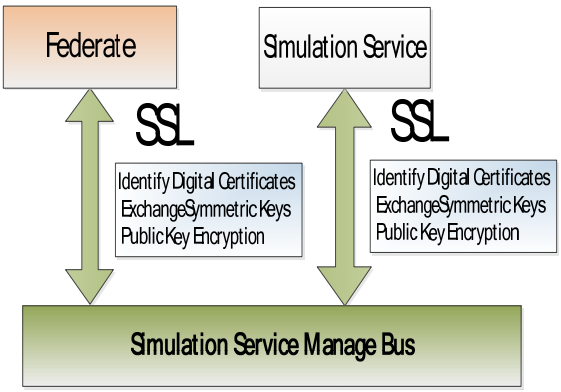

Fig .7 Communication security schematic plan

3.4 Router-Firewall Service

With the rapid development of the Internet, the security and efficiency of public networks offers a great challenge. Client and server hosts with access to public networks often reside behind protective router-firewalls that not only restrict incoming connections, but also allow the protected networks to run in a private address space using Network Address Translation (NAT). Under these circum-stances, end-to-end data transmission and communication requires the federate, simulation service, and SSMB communicates to be able to communicate through firewalls. The federates, simulation services, and SSMB communicate on a non-trusted network but are in a private network behind a firewall, which leads to many problems: (1) A dedicated port on the server's firewall must be opened and configured to forward messages to the server, (2) If the server uses multiple endpoints (e.g., to support both TCP and SSL), then a firewall port must be dedicated to each endpoint; (3) The proxies of the federate and the simulation must be configured to use the

SSMB's "public" endpoint, which is the host name and dedicated port of the firewall, (4) if the SSMB returns a proxy as the result of a request, the proxy must not contain the SSMB's private endpoint because that endpoint is inaccessible to the federate, and (5) adding a callback from the SSMB to the federate or simulation implies that the federate and simulation service will meet the same problems as SSMB. This problem is further complicated by a large number of federates and simulation services. A special Router-Firewall service is used in SDRTI to solve the above problems. This Router-Firewall service heads off requests from external proxies and retransmits the requests to the special service end, crossing the NAT/FW. In addition, the federate or simulation starts the Router-Firewall service according to their demands, and the callback request two-way connection among the federate, simulation service, and SSMB can be the existing connection among federate, simulation service, and SSMB. The working principle of the Router Firewall service is shown in Figure 8.

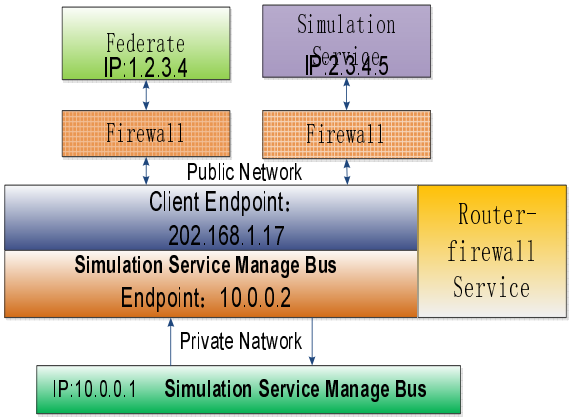

Fig.8 The working principle of the router-firewall service schematic plan

The Router-Firewall service has many merits: (1) only one front-end port is necessary to support any number of servers, (2) federates and simulation services often require only minimal changes to use the Router-firewall service, (3) the SSMB is unaware of the presence of the Router-Firewall service and requires no modifications whatsoever to use it. From the SSMB's perspective, the Router-Firewall service is just another local client, therefore the SSMB is no longer required to advertise "public" endpoints in the proxies they create, (4) the Router-Firewall supports callback from server to client sent over an existing connection from the client to the server, thereby eliminating the administrative requirements associated with supporting callbacks in the client firewall, and (5) the Route-Firewall supports the TCP and SSL protocols, which further enhance commun-ication security.

3.5 Service decoupling

Different services need to transfer data. For example, the object management service needs to obtain data from the federate management service, leading to close coupling between the management and federate services. SOA requires services to be loosely coupled as far from one another as possible. However, we use the data transfer service, an effective subscribe/publish service, to decouple the services. Publishers and subscribers obtain data by subscribing to topics, and publishers distribute data to subscribers using the data transfer service. The publishers and subscribers are transparent because of the data transfer service. We make the master and slave nodes run at the same time, with the slave node monitoring the load condition of the master node, to prevent the data transfer service from turning into a bottleneck. When the load condition of the master node is heavy, the slave node undertakes the task instead. This method can achieve load balancing between the master and slave nodes, as shown in Figure 9.

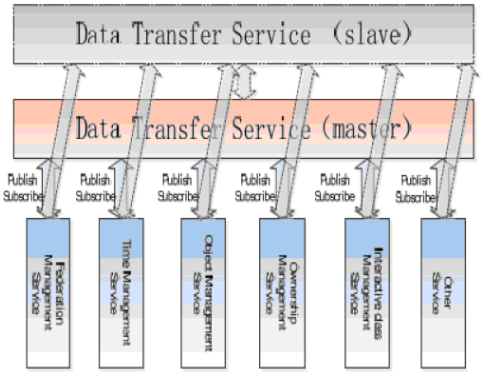

Fig.9 Data transfer service schematic plan

3.6 Communication optimization

In SDRTI, there are two message transport patterns: one-way mode and batch mode. In one-way mode, each mode is sent to the message-receiving end. In batch mode, messages are stored in the message queue and sent to the message-receiving end in batches. One-way mode emphasizes security: once the message is received, one-way mode sends the message at once, thereby minimizing the possibility of message transport error. By contrast, batch mode stores the message in a queue, thereby minimizing network overhead. SDRTI needs to transport messages among federates, simulation service, and SSMB via the network, resulting in heavy network overhead. These conditions necessitate batch mode.

4 Performance evaluations

We evaluate the performance of SDRTI in the enclosure space using Simple Collision software. The federate and simulation services behind the firewall, which are distributed via the public network, use different programming languages, such as C++, Java, and C#, as shown in Figure 10. In this simulation program, each instance represents a federate. The instance joins the federation and controls the travel route of the local ball. At the same time, the instance publishes its coordinates in

real time and subscribes the coordinates of other balls controlled by other federates. Each federate marks the location of its local ball using a red ball, and the location of other balls using black balls. The running interface of the test program is shown in Figure 11. The simulation logic of this simulation program is simple. We can change the number of simulation entities and time marching cycles in the simulation program to objectively reflect the performance of the simulation system.

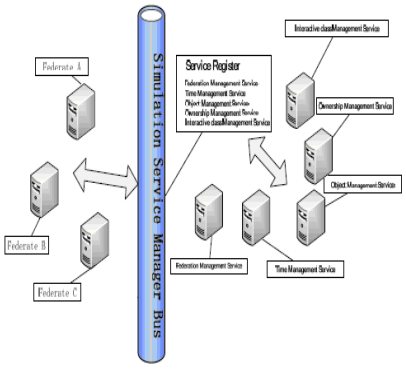

Fig . 10 Simulation deployment

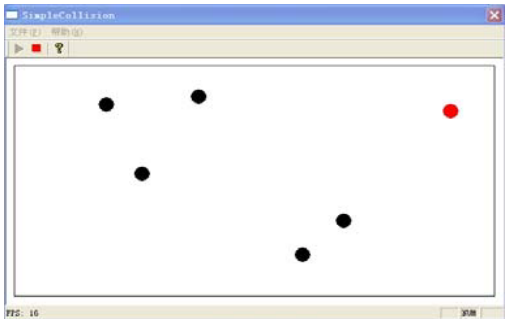

Fig.11 Simple collision

4. 1. Evaluation of the response speed

From 2 to 30 simulation entities join the federation in order and each entity (small ball) runs at its fastest speed. The delay time of each step is zero, and the speed of the actual step is absolutely dependent on the response delays of the SSMB, federates, and simulation service. We can record the actual response step per second in the simulation. First, we use the same Simple Collision program and compare the actual step between Gong Yuan Run Time Infrastructure (GYRTI) [11] and SDRTI in synchronous message transport mode, as shown in Figure 12. Second, we use the same simulation, only in the asynchronous message transport mode, as shown in Figure 13.

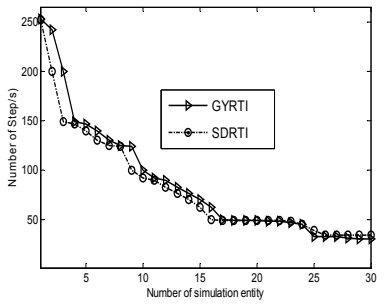

Fig .12 Actual step in synchronous mode

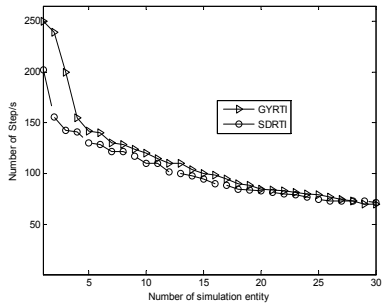

Fig.13 Actual step in asynchronous mode

4.2. Evaluation of load capacity From 1 to 100 simulation entities join the federation, and each entity (a small ball) advances 25 steps per second. The simulation record keeps track of the CPU utilization of the RTI server in the GYRTI, and the SSMB in SDRTI. The simulation results are shown in Figure 14.

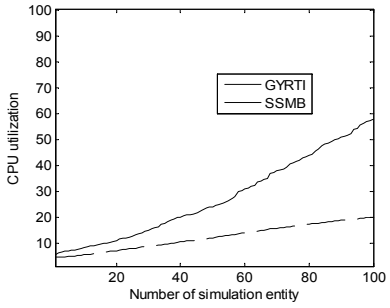

Fig .14 CPU utility

From Figure14, it can be seen that the CPU utility of the RTI server in the GYRTI increases linearly as the number of simulation entities increase. In the SDRTI, the SSMB is relieved of its load because none of the simulation services execute in the SSMB. The CPU utility of SSMB in SDRTI increases with relatively stable frequency compared with the rapidly increasing CPU utility of the server in GYRTI. The load of the RTI server is relieved by the distributed simulation service in the network. The next simulation involves communicating in batch mode and one-way mode, and comparing the network traffic of these two communication mode. The simulation results are shown in Figure 15.

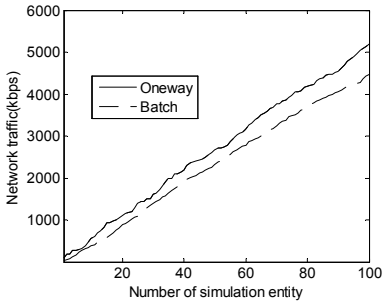

Fig.15 Network traffic

From Figure 15, it can be seen that batch mode exhibits less network traffic than one-way mode, which is important for saving network bandwidth in complex network environments.

From Figures 12 and 13, it can be seen that the advance step of GYRTI is greater than that of SDRTI at the beginning of the simulation, because GYRTI is the central management mode. However, the different decomposition simulation services in SDRTI are distributed across the network, and the responses from these simulation services cause higher delay in SORTI compared with GYRTI. However, the advance step of SDRTI exceeds that of GYRTI as the number of simulation entities increase, as the operating efficiency of GYRTI suffers due to its heavy load. In SDRTI, most of the work is done by the simulation services distributed across the network.

In the SDRTI, if the synchronous mode is used to transport messages, the response speed is fast when the number of simulation entities is low. However, the response speed decreases linearly when the number of simulation entities decreases. If the asynchronous mode is used, the response speed decreases logarithmically. Hence, we can conclude that response speed is better in asynchronous mode than in synchronous mode.

5 Conclusions

This study analyzes the problems associated with distributed simulation research based on the present situation, and points out the disadvantages of achieving web services-based service-oriented RTI. This study proposes six key technologies to achieve SDRTI, which offers several advantages, such as heterogeneous communication, load balancing, loose coupling of services, communications security, and communications optimization. A simple case study is performed based on the SDRTI. The simulation results demonstrate that:

First, SDRTI achieve load balancing by decoupling RTI services to different simulation services and reduce the burden on the SSMB. Second, SDRTI achieve communicating among different developed language and platform by the model mapping mechanism. Third, SDRTI achieve traversing NAT/FW and achieving end-to-end communication and data transmission in public networks by Router-Firewall Service. The last but not least, SDRTI achieve communication optimization by optimal communication mode. The asynchronous mode and batch mode could ensure communication stability and efficiency.

In our future work, we will improve SDRTI performance. First, to improve its stability, SSMB should have many slave nodes and the SSMB in slave nodes could work instead of the SSMB in master node when necessary. Second, to improve its flexibility, we will update the SDRTI to meet most of the HLA Evolved standards that include Modular Federate object model (Modular FOM), Evolved Dynamic Link Compatible Application Program Interface (EDLC API), Smart Update Rate Reduction (SURR).

Acknowledgement

This work was supported by National Natural Science Foundation of China (Grant no. 70791137).
The authors are grateful to the anonymous referee for a careful checking of the details and for helpful comments that improved this paper.

References

[1] Fujimoto,R.M.Parallel and Distributed Simulation Systems. Wiley Interscience, New York. (2000).
[2] Zhou yan, Dai jianwei. Simulation Program Design.Beijing: Publishing house of electronic industry,(2002).
[3] SAIC,RTI-NG l.3V3.2 Release Notes, Scienee Applications International Corp.(2000).
[4] M?K Technologies. MÄK Real-time RTI.http://www.mak. com/rti.htm,(2003) .
[5] IEEE 1516, Standard for Modeling & Simulation-High Level Architecture, (2000).
[6] Pitch AB.pRTI1516 USER GUIDE. http://www.pitch.se/pRTI1-516 /. March (2003).
[7] Mikael Karlsson, Lenart Olsson. pRTI.1516.Rationale and Design. Paper 01F-SIW-038 in Proceedings of the 2001 Fall Simulation Interoperability Workshop, http://siso.sc.ist.ucf.edu/siw/,(2001)
[8] Huang jian. Research of HLA simulation system software framework and key technology, china, Changsha: National University of Defense Technology, (2000).
[9] Huang Jian, Hao Gian Guo,Huang Ke Di. The Research and Application of HLA-Based Distributed Simulation Environment KD-HLA.Journal of System Simulation,vol.16, no.2, 214-221, (2004)
[10] Distributed interactive simulation platform BH RTI 2.2 Programming Manual-IEEE1516 standard. Beijing University of Aeronautics, (2006).
[11] Liu Bin, Zhang Hongjun, Yang Xiaojia, GY-RTI: An Integrated Distributed Simulation Environment, IEEE International Conference on Networking, Sensing and Control (ICNSC08), 232-235, (2008)

[12] Don B, Michael Z, Pullen J M, et al. Extensible Modeling and Simulation Framework Challenges for Web-based Modeling and Simulation[R]. Monterey, CA: Findings and Recommendations Report,(2002).
[13] K.L. Morse, D.L. Drake, R.P.Z. Brunton, Web enabling HLA compliant simulations to support network centric applications, in: Proceedings of the 2004 Symposium on Command and Control Research and Technology, San Diego, 594-559, (2004).
[14] MorseK LDrke D LBrtunton R PWeb Enabling an RTI-an XMSF profile//procOf the 2003 European Sinmulation lnteroperebility workshopJune (2003).
[15] Möller BLöf SMixing Service Oriented and High Level Architectures in Support of GIG Procof the 2005 Fall Simulation Interoperability Workshop.(2005).
[16] Möller BLöf SA Management Overview of the HLA Evolved web Service API//Procof the 2006 Fall Simulation Interoperability Workshop.(2006).
[17] Möller B,Dahlin CA First Look at the HLA Evolved web Service API//Procofthe 2006 European Simulation Interoperability Workshop.(2006)
[18] Monica Dragoicea, Laurentiu Bucur,Wei-Tek Tsai, Hessam Sarjoughian.Integrating HLA and Service-Oriented Architecture in a Simulation Framework. 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. 861-866 (2012).
[19] Zhang WeiZhang TongZha Yabin. Web Service Enabling of HLA-based Distributed Simulation. JOURNAL OF NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY, vol.30,no.5, 120-124,(2008)
[20] Zhang Wei, Zha Ya-bing. Research on Key Technologies of Web Enabled RTI. Journal of System Simulation, vol.20, no.3, 6414-6417, 6437(2008)
[21] Jiang Zhao-jin, Huang Jian, Huang Ke-di. HLA/RTI Simulation System Extension Based on Web Services. Advanced Manufacture and Management, vol.27, no.1, 39-40,51, (2008).
[22] Zhou Xin, Wei Jun-hu, Li Peng, Su Qin. Design and Implementation of Distributed RTI Based on Web Services. Journal of System Simulation, vol.20, no.8, 2064-2067, (2008)
[23] Shucai Tang, Tianyuan Xiao, Wenhui Fan. A collaborative platform for complex product design with an extended HLA integration architecture. Simulation Modelling Practice and Theory. 18 , 1048-1068,2010
[24] H. Wang, H. Zhang, An integrated and collaborative approach for complex product development in distributed heterogeneous environment , International Journal of Production Research. vol.46, no.9, 2345-2361, (2008)
[25] Heming Zhang, Hongwei Wang, David Chen, Gregory Zachar-ewicz. A model-driven approach to multidisciplinary collabora-tive simulation for virtual product development. Advanced Engi-neering Informatics,24, 167-179, (2010).
[26] Mark Turner, Fujun Zhu, Ioannis Kotsiopoulos, Michelle Russell, David Budgen, Keith,Bennett, Pearl Brereton, John Keane, Paul Layzell and Michael Rigby. Using Web Service Technologies to create an Information Broker: An Experience Report. Proceedings of the 26th International Conference on Software Engineering. IEEE Computer Society Press, 831-839 (2004)
[27] James Byrne, Cathal Heaveya, P.J.Byrneb. A review of Web-based simulation and supporting tools. Simulation Modelling Practice and Theory. 18, 253-276, (2010).
[28] Baohua Jin, Dong Yaozou. Research of Web Service based on P2P. 2010 Second International Conference on Communication Software and Networks. 412-415 (2010).
[29] Deng Ziyun. SOA practitioners : system integration in a distributed environment.Publishing House of Electronics Industry, 12,2010.
[30] M. Turner, D. Budgen and O.P. Brereton, "Turning Software into a Service", IEEE Computer, 38-44 (2003).
[31] Boyens C, G nther O. Trust is not enough: Privacy and security in ASP and Web service environments. In: Manolopoulos Y, et al., eds. Proc. of the 6th East European Conf. on Advances in Databases and Information Systems. Bratislava: Springer-Verlag, 8-22 (2002).
[32] Thelin J, Murray PJ. A public Web services security framework based on current and future usage scenarios. In: Arabnia H, eds. Proc. of the Int'l Conf. on Internet Computing (IC2002). Las Vegas: CSREA Press, 825-833 (2001)
[33] Jameela Al-Jaroodi, Nader Mohamed. Service-oriented middle-are: A survey. Journal of Network and Computer Applications. 35, 211-220(2008)
[34] Henning M, Spruiell M, Distributed Programming with Ice, http://www.zeroc.com/, (2011)
[35] Michael P.Papazoglou. Web Services Principles and Technology. China Machine Press. May(2011).
[36] Xu Lijuan, PENG Xiaoyuan. HLA-based Simulation Service Bus Research. Journal of System Simulation, vol.18, no.2, 347- 349, (2006).
[37] Di Shiqiang, MIANXIANG FUWU DE JIANMO YU FANGZHEN JISHU.National defense industry press,(2011).
[38] D.Wu, X.J.Ban, F.Oquendo.An Architecture Model of Distributed Simulation System Based on Quotient Space. Applied Mathematics & Information Sciences.6,No. 2,603-609 (2012)
[39] Heng He, Ruixuan Li, Xinhua Dong, Zhi Zhang, Hongmu Han.An Efficient and Secure Cloud-Based Distributed Simulation System. Applied Mathematics & Information Sciences. 6, No. 3, 729-736 (2012)
[40] Wenya Tian. Design and Implementation of Web-Based Food Regulatory Information Resources Management Platform. Applied Mathematics & Information Sciences. 5, No. 2, 105-111 (2011)

Zhiteng Wang was born in Heilongjiang province in 1982, china. He received the master's degree in Nanjing artillery academy in 2010. Currently, He is a doctorial student in Technology Military Simulation Technology and data engineering Lab in PLA University of Science. He has joined in many scientific research items on distributed communication. He research on Service-Distributed Run-time Infrastructure.

Hongjun Zhang is a chief professor and doctoral supervisor at PLA University of Science and Technology. His researches areas are military simulation, model and military operational research, and system integration.

Rui Zhang received Ph. D. on communication engineering at school of PLA University of Science and Technology. He is currently an adjunct professor in PLA University of Science and Technology. His researches area is simulation engineering.

Yong Li is a professor and doctoral supervisor at Nanjing Artillery Academy. His researches areas are science of tactics and military education and training.

Baoyu Xu was born in Shenyang province in 1974. He is a doctorial student in Technology Military Simulation Technology data engineering Lab in PLA University of Science. He research on Distributed simulation.