Jong-Hyun Kim · Kangsun Lee
Satoshi Tanaka · Soo-Hyun Park
Editors

Proceedings in Information and Communications Technology

PICT 4

Advanced Methods, Techniques, and Applications in Modeling and Simulation

Asia Simulation Conference 2011, Seoul, Korea, November 2011, Proceedings

Springer

Proceedings in
Information and Communications Technology 4

Jong-Hyun Kim   Kangsun Lee   Satoshi Tanaka
Soo-Hyun Park (Eds.)

Advanced Methods, Techniques, and Applications in Modeling and Simulation

Asia Simulation Conference 2011
Seoul, Korea, November 2011
Proceedings

Springer

Volume Editors

Jong-Hyun Kim
Yonsei University
Korea
E-mail: jhkim34@yonsei.ac.kr

Satoshi Tanaka
Ritsumeikan University
Japan
E-mail: stanaka@media.ritsumei.ac.jp

Kangsun Lee
MyongJi University
Korea
E-mail: ksl@mju.ac.kr

Soo-Hyun Park
Kookmin University
Korea
E-mail: shpark21@kookmin.ac.kr

ISSN 1867-2914 e-ISSN 1867-2922
ISBN 978-4-431-54215-5 e-ISBN 978-4-431-54216-2
DOI 10.1007/978-4-431-54216-2
Springer Tokyo Heidelberg New York Dordrecht London

Library of Congress Control Number: 2012946923

CR Subject Classification (1998): I.6

Typesetting: Camera ready by author and data conversion by Scientific Publishing Services, Chennai, India.

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The AsiaSim2011 (Asia Simulation Conference 2011) was held November 16–18, 2011, in Seoul, Korea. The Asia Simulation Conference, held annually, is organized by ASIASIM (Federation of Asian Simulation Societies), KSS (Korea Society for Simulation), CASS (Chinese Association for System Simulation), and JSST (Japan Society for Simulation Technology) and provides a forum for scientists, academicians, and professionals from the Asia-Pacific region and from other parts of the world. Participants present their latest exciting research findings in various fields of modeling, simulation, and their applications.

This volume is the proceedings of AsiaSim2011. At the conference, 61 papers, stringently selected, were presented and three keynote speeches introduced new simulation technologies and research trends. Full-length versions of all submitted papers were refereed by the international program committee, with each paper receiving at least two independent reviews. This volume publishes papers selected from among those presented at the conference.

In addition to the scientific papers presented, the conference featured keynote talks by three invited speakers: Axel Lehmann (Bundeswehr University, Munich, Germany), Ichiro Hagiwara (Tokyo Institute of Technology, Japan), and Bo Hu Li (Beijing University of Aeronautics and Astronautics, China). We are grateful to them for accepting our invitation and for their talks. We also would like to express our gratitude to all contributors, reviewers, and program committee and organizing committee members who made the conference very successful. Special thanks are due to Soo-Hyun Park, Yun-Bae Kim, and Kangsun Lee, Chairs of the Program Committee, Organization Committee, and Publication Committee of AsiaSim2011, for their hard work in various aspects of conference organization.

Finally, I would like to acknowledge the partial financial support of the Korea Federation of Science and Technology Societies. I also would like to express appreciation for the publication support from Springer Japan.

November 2011

Jong-Hyun Kim
General Chair of AsiaSim2011

Organization

Honorary Chairs

Chan-Mo Park	POSTECH, Korea
Seong-Joo Park	KAIST, Korea
Doo-Kwon Baik	Korea University, Korea
Bo Hu Li	Beijing University of Aeronautics and Astronautics, China
Toshiharu Kagawa	Tokyo Institute of Technology, Japan

General Chair

Jong-Hyun Kim	Yonsei University, Korea

General Co-chairs

Jin-Woo Park	Seoul National University, Korea
Tag Gon Kim	Korea Advanced Institute of Science and Technology, Korea
Myoung Hee Kim	Ewha Womans University, Korea
Axel Lehmann	Bundeswehr University, Munich, Germany
Osamu Ono	Meiji University, Japan
Koji Koyamada	Kyoto University, Japan
Qinping Zhao	Beihang University, China
Kitti Tirasesth	KMITL, Thailand
Marzuki Khalid	UTM, Malaysia
Bernard P. Zeigler	University of Arizona, USA

Organizing Committee

Chair

Yun Bae Kim	Sungkyunkwan University, Korea

Dong-Won Seo	Kyung Hee University, Korea
Zhang Lin	Beihang University, China

International Program Committee

Chairs

Sung-Do Chi	Hankuk Aviation University, Korea
Soo-Hyun Park	Kookmin University, Korea
Kangsun Lee	Myongji University, Korea
Satoshi Tanaka	Ritsumeikan University, Japan

Hyung-Jong Kim	Seoul Women's University, Korea
Tianyuan Xiao	Tsinghua University, China
Noriyuki Komine	Tokai University, Japan
Jung Gil Choi	Korea Institute of Industrial Technology, Korea
Norman Foo	University of New South Wales, Australia
Sumit Ghosh	University of Texas at Tyler, USA
David Goldsman	Georgia Institute of Technology, USA
Alemdar Hasanoglu	Kocaeli University , Turkey
Farnando J. Barros	University of Coimbra, Portugal
V. Kachitvichyanukul	Asian Institute of Technology, Thailand
Yukio Kagawa	Nihon University, Japan
Ki-hyung Kim	Ajou University, Korea
Haklin Kimm	East Stroudsburg University, USA
Hessam S. Sarjoughian	University of Arizona, USA
Roy Sato	University of Tsukuba, Japan
Mi Ae Shon	Sungkyunkwan University, Korea
Shingo Takahashi	Waseda University, Japan
Xiao Tianyuan	Tsinghua University, China
Peng Xiaoyuan	Beijing University, China
Guang Leng Xiong	Tsinghua University, China
Mohd Nazmi Yosoff	International Islamic University, Malaysia

Publication Committee

Chair

Seong Yong Jang	Seoul National University of Science and Technology, Korea
Hee Seok Seo	Korea University of Technology and Education, Korea

Industrial Committee

Chair

Dug Hee Moon	Changwon National University, Korea

Table of Contents

Modeling Methodology (II)

Communication Networks and Simulations

Simulation in General Engineering (I)

Simulation in General Engineering (II)

Simulation in General Engineering (III)

Military Simulation (I)

Military Simulation (II)

Military Simulation (III)

Simulation Methods and Tools

Simulation and Visualization

Flight Simulation on Tiled Displays with Distributed Computing Scheme

Sangwon Chae[1], Soon-Bin Yim[2], and Youngshin Han[2,*]

[1] University of California, Irvine,
Irvine, CA 92697, U.S.A.
chaes@uci.edu
[2] Sungkyunkwan University
School of Information and Communication Engineering,
Suwon, Gyeonggi-do 440-746, South Korea
{sbyim,hanys}@skku.edu

Abstract. Flight simulator is based on real aircrafts. The main purpose of flight simulator is to prevent accidents when people control real flight through the training with various unexpected scenarios during flight. It is difficult to handle large scale geographical data on ordinary personal computer. Therefore, we propose distributed flight simulation which provides wider screen view on tiled displays and high system performance with distributed computing environment.

1 Introduction

Simulation is based on the imitation of real things, state of affairs or process. It shows similar characteristics of real objects such as aircraft, generator, analog/digital circuit and so on. Purposes of those simulators are various such as research, drill, and entertainment[1]. Applications of simulators which show the simulation of real objects are war game, aero system, factory automation and so on. For educational purposes, we can simulate on personal computers, it is more efficient to run simulators on tiled display which consists of multiple displays and computing nodes. Tiled displays use a diversity of implementation immersed virtual reality(VR). VR which is implemented on tiled displays provides wide display environments, so it is possible for developers to co-work. Tiled displays are efficient to not only face to face VR which multiple users share one system to communicate and work together, but also remote VR which small group users can share their works through network. Moreover, if we run simulators on distributed computing environments, we can achieve scalability to handle large data and stability to run on real time. It is an important issue to show the characteristics of real objects on real time.

In this paper, we target a flight simulator which a user can control virtual flight objects in order to get used to handling real ones. Indeed, we run a flight simulator on tiled displays which provide distributed computing environments. That is, the

* Corresponding author.

J.-H. Kim et al. (Eds.): AsiaSim2011, PICT 4, pp. 1–6, 2012.

purpose of our research is for users/pilots to understand various scenarios of battle, interceptions, and evasion more effectively through our flight simulations. In addition, we can provide wider and larger screen view to users on tiled displays and handle large dataset through distributed environments.

This paper describes our proposed algorithm in Section 2. Section 3 shows our system overview and Section 4 indicates the results of our system. Section 5, which concludes, summarizes our paper and discusses the future works of our system.

2 Algorithm

In this section, we describe our proposed algorithms. This simulator consists of three parts, a simulation engine, geographical database, and a rendering engine. We generate random geographical data and implement simulation engine and rendering algorithms.

2.1 Simulation Engine

Both simulation and rendering engine of our system have functions to save and load both of input and output data. Simulation engine provides various input methods such as joysticks, keyboards, and mouse for users to control virtual aircraft and saves data of their flights. In addition, we can see pre-saved data to see educational scenarios without any control.

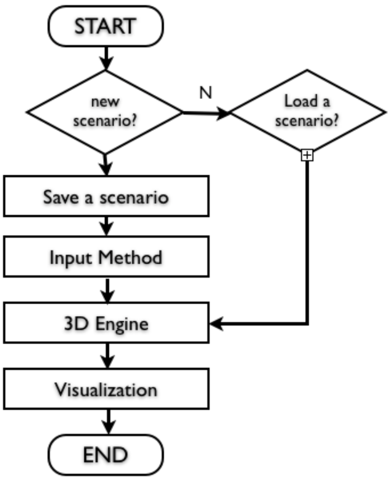

Fig. 1. It shows a simple simulation engine flowchart. We can choose whether it is new scenario to create or old scenario to load. We can control virtual crafts through input method and see the result on displays.

2.2 Distributed Rendering Algorithm

Display wall system is shown recently[4], these walls consist of one or multiple displays which are connected multiple computing nodes. In this environment, it is important how to distribute data to each computing node for rendering.

As you can see, it consists of two parts, one is a control node and the other is computing display node. A main purpose of control node is to collect and distribute all data to each computing display node. It is difficult to handle large data on one node, so it divides an entire data into pieces of data. Since geographical data and aircrafts' information are based on positions, it is available to divide a whole data into sub-dataset. In addition, a main purpose of computing displays nodes is to visualize all data on their displays after rendering. Each computing display node communicates to a control node for sending and receiving dataset after processing its procedures.

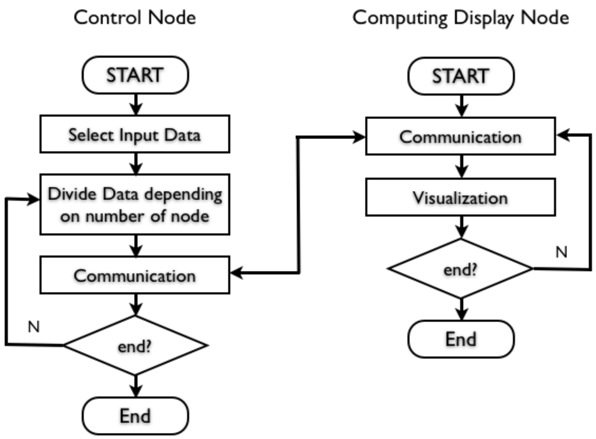

Fig. 2. It shows a simple distributed rendering engine flowchart. It consists of two parts, a control node which control geographical data and flight information and computing display nodes which display all data on each display.

3 System Overview

Our system consists of two parts, a control node and compute-displaying nodes. Figure 3 shows the system overview of our research we discuss in Section 2. Figure 4 shows Highly Interactive Parallel Display Wall(HIPerWall)[5]. Our system consists of 50 displays and 25 compute-displaying nodes. Its resolution is 200M pixel at once. It supports to display very large picture without any zoom or compression of data. When we simulate virtual aircrafts on HIPerWall, it is possible to understand scenarios more effectively and detail.

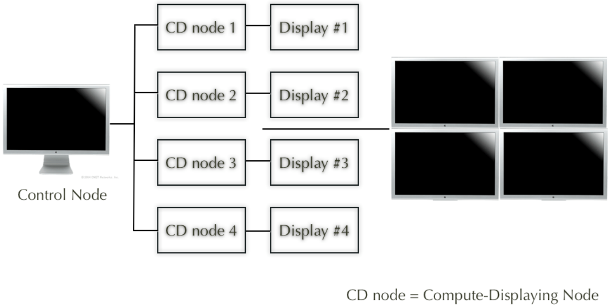

Fig. 3. There is a control node to collect and distribute data to each compute-displaying node. Each compute-displaying node is connected to one or two display(s). They are visualizing data set for users to see a scenario of simulation.

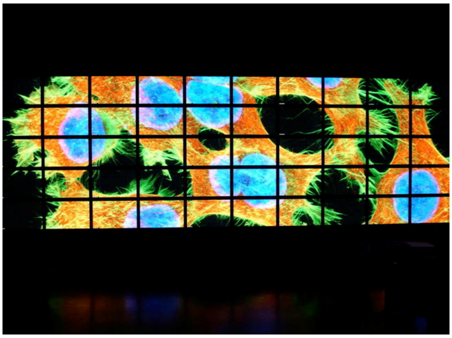

Fig. 4. Highly Interactive Parallel Display Wall(HIPerWall) consists of 25 compute-displaying nodes and 50 mac cinema displays. Left picture shows an example of visualizing medical information and right picture shows the back of HIPerWall.

4 Result

In this section, we show the simulation results on a single display and tiled displays. In Figure 5, the result on tiled displays consists of four displays and four compute-displaying nodes(lower picture) shows more details and higher quality rather than the result on a single node and a single display(upper picture). When practical data is applied, it is possible to have more complicated and larger geographical data to visualize. It compromises the overall system performance such as processing time and visualization. Tiled displays provide users to have wider views with higher resolution to display large geographical data at once.

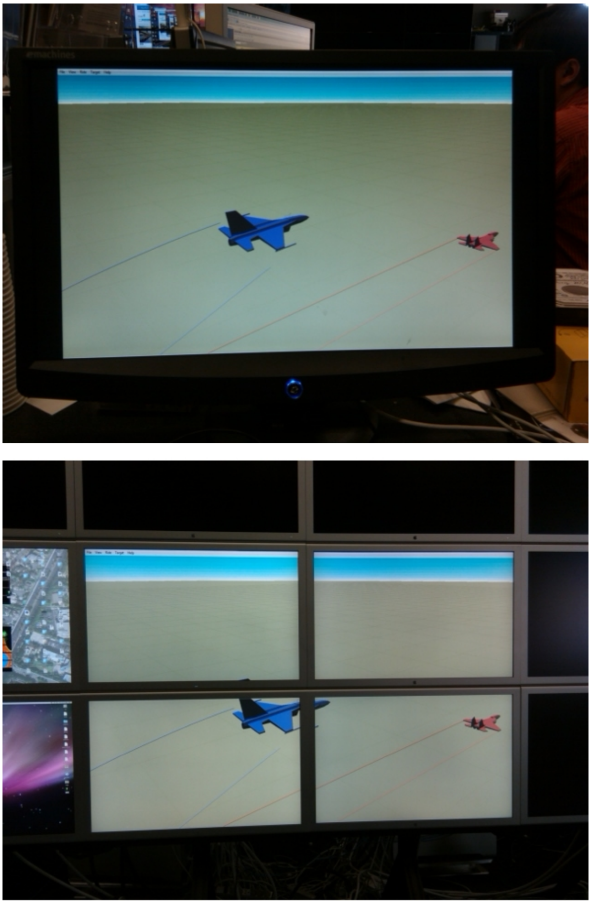

Fig. 5. There are the results on a single display and compute-displaying node(upper) and tiled displays which consists of four displays and four compute-displaying nodes(lower). Tiled display provides more detailed and higher quality results to users.

5 Conclusion and Future Works

We implemented distributed flight simulation on tiled displays. It provides more detailed results to users with higher resolutions and it is scalable on tiled display with distributed computing scheme.

However, there are several issues of tiled displays. Because of the frames of each display, it is difficult to discern objects when multiple objects go through panels. Moreover, it is expensive to build entire display walls. We can build projector based display wall instead of displays, but the cons of projector based wall are lower resolution and a keystone issue.

Major component of our future work is the implementation of the module to provide practical geographical data and more detailed aircraft. Also, we will implement the module to control multiple aircrafts from multiple input methods in real-time environment. It can show educational scenarios of large scale battles or training on tiled displays.

Acknowledgement. This work is supported by Basic Research Program through the National Research Foundation of Korea(NRF) funded by Ministry of Education, Science and Technology(2011-0379-000).

References

1. Choi, S., Chae, S., Han, Y., Lee, C.: 3D Flight simulator for flying tactics. In: Korean Simulation Conference, pp. 37–42 (2003)
2. Park, K., Park, C.: A research of flight simulator' components. In: Korean Aeronautical and Space Sciences Conference, pp. 239–245 (November 2001)
3. Kim, J., You, C.: Light aircraft simulator on personal computer. In: Korean Aeronautical and Space Sciences Conference, pp. 392–397 (April 1994)
4. Mueller, C., Gregor, D., Lumsdaine, A.: Distributed Force-Directed Graph Layout and Visualization. In: Eurographics Symposium on Parallel Graphics and Visualization (EGPVG 2006) (May 2006)
5. HIPerWall, `http://hiperwall.calit2.uci.edu/`
6. Ratan, K., Jeff, L.: Flight Simulator Evaluation of a Cockpit Display of Traffic Information (CDTI) With and Without Indications & Alerts (IA). Honeywell Contract Report to FAA, Redmond (2010)

A Novel Federation Development and Execution Process Based on Collaboration Ontology

Hongbo Sun, Wenhui Fan, Tianyuan Xiao, Yanbing Liu, and Yueting Chai

National CIMS Engineering Research Centre, Tsinghua University, Beijing 100084,China
hsun@ieee.org

Abstract. This paper presents a novel Federation Development and Execution Process aiming to employ collaboration ontology as mutual understanding of High Level Architecture. The proposed approach includes three main stages: Conceptual Analysis, Collaboration Preparation and Federation Execution. It uses collaboration ontology to represent collaboration knowledge such as conceptual model, domain knowledge, SOM, FOM, FED/FDD files, class RootObject in a consistent way. This knowledge representation method improves semantic completeness and transformation efficiency of collaboration knowledge and reduces workload without the loss of accuracy and consistency. This paper also discusses some ontology based collaboration knowledge modeling, manipulating, organization and implementation issues on the basis of collaboration ontology, which includes FCA modeling method, extending UML sequence diagram, ontology fusion, ontology maintenance and ontology template. The proposed approach has great potential to improve efficiency of FEDEP, reduce the work load for adaptive adjustment of ever-existing platforms, and enhance the applicability and flexibility of HLA systems.

1 Introduction

As a well-known modeling and integration standard of distributed simulation, HLA (High level Architecture [1]) has been successfully adopted in various simulation systems, and extended to some other research areas [2]. Since simulation is an important part of collaborative product development, some CPD (Collaborative Product Development) systems use HLA as their basic architecture [3,4].

However, when applying HLA in this research area, there raise some new challenges. Within these challenges, mutual understanding (collaboration) knowledge modeling and manipulating among several subsystems is an essential issue to support solutions of other issues.

Fortunately, ontology in knowledge engineering is the semantic basis of communication among domain entities. It is applicable to automatic reasoning, knowledge representation and reuse [5].

Sun H. has proposed the concept of collaboration ontology to solve mutual understanding among several independent systems [6]. Because the algebraic system defined on the concept set of HLA collaboration and the partial order relations of these

J.-H. Kim et al. (Eds.): AsiaSim2011, PICT 4, pp. 7–14, 2012.

concepts have the same upper bound and lower bound, it can be deemed as a concept lattice [7]. The collaboration ontology is formally defined as follows:

$$O ::= (C, H_C, R_C, H_R, M, R_M, A)$$

Collaboration ontology O is defined as a seven tuple. C denotes collaboration concept set of HLA collaboration. H_C defines a set of partial orders on concept set C, which give the inherit relations among the concepts involved. The concepts set and inherit relations defined on that set form a Directed Acyclic Graph (DAG) whose source is the given model of collaborative product and whose sink is binary fragments. R_C denotes a set of non-inherit partial order relations on concept set C, corresponding to concept attributes. H_R defines inherit relations on partial order relation set R_C . M is a series of meta ontology concepts, which give a series inheritable instances of R_C. R_M denotes a set of partial order relations under M, which describe the relations among elements in meta ontology set, and are also the basis for ontology reasoning. A defines a set of axioms among ontology concept set and meta ontology relation set, which provide the major premises of ontology reasoning.

This paper proposes a collaboration ontology based FEDEP (FEderation Development and Execution Process) to give an outline of how to use, organize and manipulate collaboration ontology to meet various requirements. This paper is organized as follows: the motivation of this research is described in Section 2; collaboration ontology based FEDEP is proposed in Section 3; discussions and conclusions are given in the last section.

2 Motivation

In HLA-based federation development, collaboration knowledge is mainly manipulated as Fig. 1 shows [8], and can be divided into 11 steps: Analyze Scenario, Generate Federation Conceptual Model, Analyze Collaboration Requirements, Generate FOM, Negotiate Federation Agreements, Realize Federate, Realize RTI, Create Federation, Start Federation Execution, Execute Federation and Terminate Federation Execution.

In this process, the input is federate SOMs, Existing Conceptual Models and collaboration Aim. The output is Scenario Instances, Federate Modification, Supporting Database, RTI & its initial data, interaction/object class instance, class subscriber, class publisher, synchronize point and results. Resources involved include Scenario Lib, Domain Resources, Data Dictionary, OM Lib and Other Resources. Scenario Lib supports *Analyze Scenario* by *Existing Scenarios.* Domain Resources provides *Domain Knowledge* to analyze scenario and generate collaboration conceptual model. Data Dictionary stores Meta OM models for Generate *FOM.* When *Generate FOM*, related *Existing FOMs and BOMs* also need to be retrieved from OM Lib. Other Resources include tools and knowledge related to *Generate FOM.*

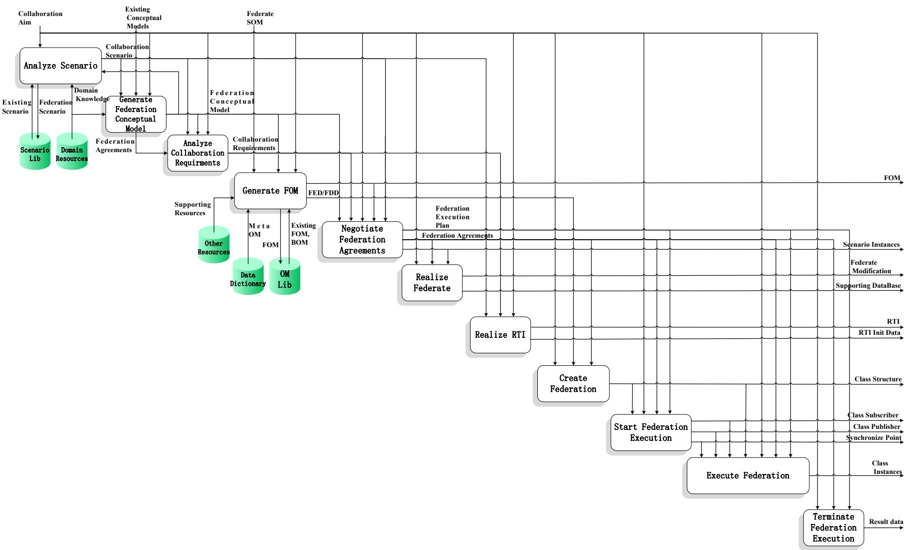

Fig. 1. Collaboration knowledge modeling and usage in FEDEP

Collaboration knowledge also represents itself as Collaboration Scenario, Federation Conceptual Model, Federation Execution Plan, FED/FDD, FOM, Collaboration Requirements, Class Subscriber, Class Publisher, and Class Structure. Collaboration Scenario describes environment and desired process of collaborations. Federation Conceptual Model is the description of concepts and their relations in collaborations. Federation Execution Plan gives operation order of participant federates. FOM is the standard description of federation concepts. FED/FDD stores FOM information. Collaboration Requirements describes *Federation Execution Plan* and collaboration requirements of federates. Interaction/object Class Structure is inheritance relations among these classes. Class Subscriber is federates that subscribe interaction or object classes. Class Publisher is federates that publish interaction or object classes, which supervise data distribution during collaborations.

From the view of collaboration knowledge, HLA semantic representation, transformation, and organize have several shortcomings which limits its consistency and efficiency.

First, collaboration knowledge takes different forms according to separated steps, which include conceptual models, domain knowledge, SOM, FOM, FED/FDD files and class RootObject. This will lead to semantic loss and incorrect delivery in the process of transformation. Moreover, it is a time-consuming and cost task to transform collaborative knowledge among different steps, and semantic accuracy and consistency are also hard to guarantee.

Second, SOM generation is not mentioned in FEDEP, which is the basis for FOM construction.

Third, class RootObject is the only one form of collaboration knowledge which can be used during collaboration execution. However, it is loaded only once before *Start Federation Execution*, so it cannot satisfy collaboration requirement changing in the process of federation execution.

In this paper, a collaboration ontology based FEDEP is proposed to give a consistent way of collaboration knowledge representation, organization and manipulating so that collaboration knowledge can be used in a more consistent, complete, flexible and efficient way.

3 Collaboration Ontology Based FEDEP

In collaboration ontology based FEDEP, corresponding to SOM and FOM, Federate collaboration ontologies describe collaboration capability and requirements of federates, federation collaboration ontology describes collaboration knowledge of federation respectively. Modeling collaboration knowledge is reflected in the creation of collaboration ontologies, manipulating collaboration knowledge corresponds to dynamical maintenance of federation collaboration ontology. Federation collaboration ontology is an important identification and data exchange basis of given federation.

The process of modeling and utilizing collaboration knowledge can be separated into three stages: Conceptual Analysis, Collaboration Preparation and Federation Execution (Fig. 2).

In stage Conceptual Analysis, scenario and domain knowledge is converted to federation collaboration scenarios which are represented by extended UML sequence graphs under the control of *Collaboration Aim*. *Collaboration Requirement* is analyzed and used as the input of collaboration preparation.

In stage Collaboration Preparation, federation agreements and federation collaboration ontology are obtained according to federate SOMs, existing collaboration ontologies, ontology template(including common collaboration ontology schema, basic information, data type transform rules), related bridge equivalence(mutual exclusion) axioms, collaboration requirements and federation collaboration scenario.

In stage Federation Execution, collaboration is executed according to federation agreements, federation collaboration ontology, collaboration requirements and collaboration aim.

3.1 Conceptual Analysis

The input of Conceptual Analysis is *Collaboration Aim*, the output is Federate Collaboration Scenario Described by Extended UML Sequence Diagram, and the resources involved include Scenario Lib, Domain Resources and Collaboration Ontology Lib. Scenario Lib provides the existing scenarios supporting scenario analysis and stores analysis results, Collaboration Ontology Lib supports collaboration requirements analysis by existing federate collaboration ontologies, and Domain Resources provide domain knowledge. This stage can be divided into two activities: Analyze Scenario and Analyze Collaboration Requirements (Fig. 3).

Analyze Scenario generates federate collaboration scenarios described by extended UML sequence diagram according to existing scenarios and domain knowledge under control of the collaboration target.

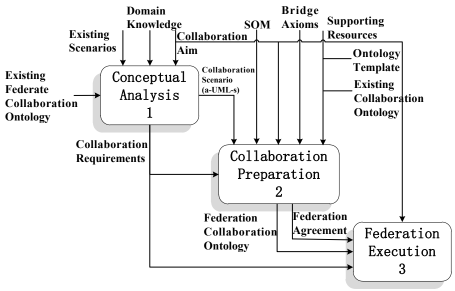

Fig. 2. Top-Level View

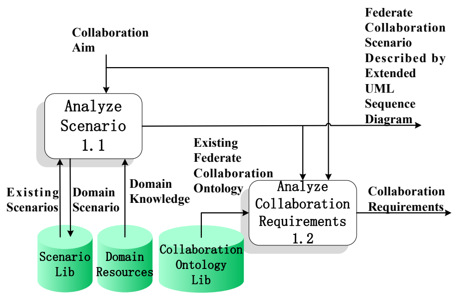

Fig. 3. Concept Analysis

Analyze Collaboration Requirement analyzes existing federate collaboration scenarios, existing federate collaboration ontology and collaboration aim, and provides collaboration requirement for future use .

3.2 Collaboration Preparation

In this stage, the input is Federate Collaboration Scenario Described by Extended UML Sequence Diagram, SOM, Collaboration Aim, Bridge Axioms, and Collaboration Requirements. The output is Federation Collaboration Ontology, Federation Agreements, Federate Modifications, Supporting Database, ORTI (Ontology-based RunTime Infrastructure, in this research ORTI is reached by add Protégé container to TH_RTI), and ORTI initial Data. And the reusable resources involve Domain Resources, Ontology Template and Collaboration Ontology Lib (Fig. 4).

In collaboration ontology-based FEDEP, federate collaboration ontology corresponds to SOM, federation collaboration ontology corresponds to FOM. Federate collaboration ontology can be established by either automatic transformation from SOM files or FCA-like method with the help of Extended UML Sequence Diagram. Federation collaboration ontology is automatically generated by federate collaboration ontologies fusion [6].

The process of Collaboration Preparation can be separated into 5 activities: Generate Federate Collaboration Ontology, Generate Federation Ontology, Negotiate Federation Agreements, Realize Federates, and Realize RTI.

Federate collaboration ontology is generated by reference basic schema, axioms and basic elements of ontology template, domain axioms of domain resources, and existing federate collaboration ontologies.

Based on federate collaboration ontologies, bridge axioms, basic schema and axioms and basic elements in ontology template, the federation collaboration ontology is fused under the control of *Collaborative Aim*.

According to federation collaboration ontology and collaboration requirement, federation agreements are generated under the guidance of *Collaboration Aim*.

Under the control of *Collaboration Aim* and *Requirements*, according to *Federation Agreements* and federate collaboration ontologies, federates and RTI can be realized properly.

3.3 Federation Execution

The input of Federation Execution is Collaboration Aim, Federation Collaboration Ontology, Federation Agreements, and Collaboration Requirements. The output is concept subscriber, publisher, synchronize points, concept instances and result data. In this process, class RootObject is replaced by federation collaboration ontology, class instances substituted by its concept instances. When collaboration requirement is changed, an online approach is trigged to fit the changes by so-called ontology maintenance. This process involves five activities: Create Federation, Start Federation Execution, Modify Collaboration Requirements, and Terminate Federation Execution (Fig. 5).

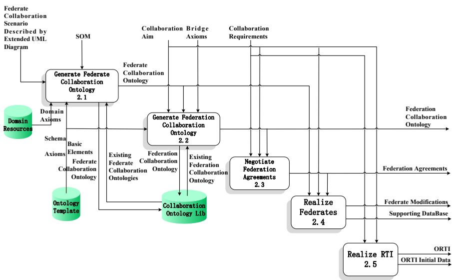

Fig. 4. Collaboration Preparation

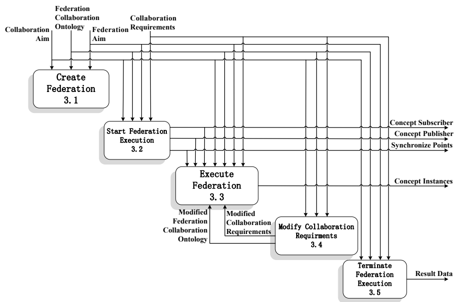

Fig. 5. Federation Preparation

Federation is created by one federate and other federates will join in one by one. The main work is to specify federation name, federation execution name, related federates and federation collaboration ontology handle.

The purpose of Start Federation Execution is to create the conceptual instances of synchronization points, register publishers and subscribers of concept instances which correspond to interaction and object classes.

In Execute Federation, data distribution is controlled by instances of publishers and subscribers. Execution process is guided by execution plan, federation agreements and collaborative target. The outputs in this stage are concept instances, update and reflection of property values.

In the process of the federation execution, collaboration requirements can be changed because of uncertainty of collaborative product development. For example, before knowing the detail of optimization requirements, optimization federate cannot tell whether it needs to ask help from simulation federates. Under this circumstance, a dynamic federation execution plan adjustment mechanism is of great importance. Its main task includes collaboration requirement modification and federation collaboration ontology maintenance.

Terminating Federation Execution stops execution of given federation and complete results collection according to Federation Execution Plan, Federation Agreements and Collaboration Aim.

From the processed described above, collaboration ontology based FEDEP can be described as Fig. 6. In this process, collaboration ontology represents collaboration knowledge such as conceptual model, domain knowledge, SOM, FOM, FED/FDD files, class RootObject in a consistent way. This knowledge representation method improves semantic completeness and transformation efficiency of collaboration knowledge and reduces workload without the loss of accuracy and consistency.

Started with analyzing federates collaboration background by extended UML sequence graph, then add appropriate semantic referring to ontology template and domain knowledge, federate collaboration ontologies could be well established by automatic transformation from SOMs or its formal background.

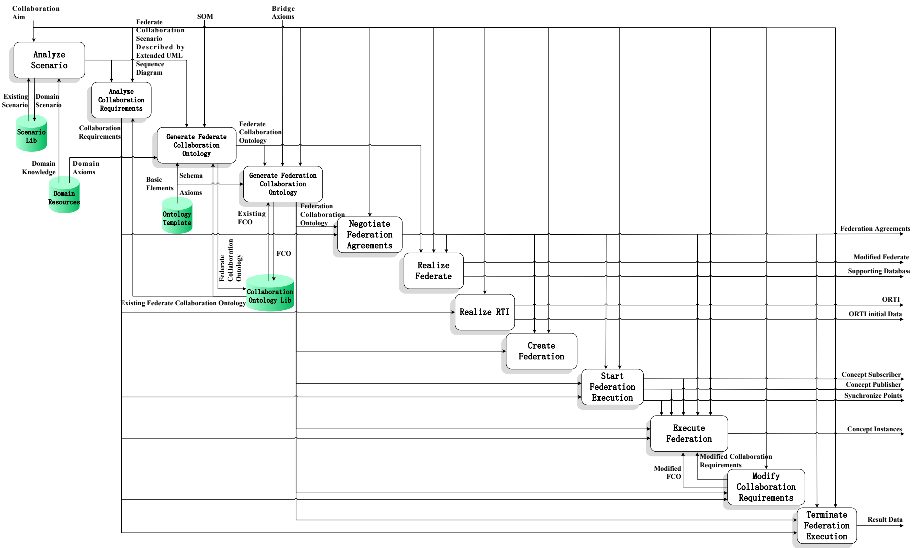

Fig. 6. Global view of collaboration ontology based FEDEP

In federation execution, collaboration knowledge is stored in federation collaboration ontology and its instances. When collaboration requirement changing, collaboration requirement modification could solve changes in an on-line manner.

4 Conclusion

Since collaboration knowledge of HLA federations adopts different representations like conceptual model, domain knowledge, SOM, FOM, FED/FDD files, and RootObject class, semantics always loses or misunderstands in their translations. In this paper, collaboration ontology describes the collaboration knowledge, and supports HLA federation development in a consistent way.

Other than collaboration knowledge is transformed ambiguously and inconsistently by negotiations which is time-consuming and cost much, this paper analyses and creates federate ontology on the basis of ontology template by extended UML sequence diagram, or uses an automata to transform SOM files to federate collaboration ontologies, constructs federation collaboration ontology by fusion of federate ontologies. These construction methods contain more semantic information, improve collaboration modeling efficiency.

Compare to HLA keep a blind eye on collaboration requirements changes, this paper introduces a novel ontology modification mechanism which employs federation ontology to substitute RootObject class, and concept instances of federation ontology to replace class instances in HLA federation executions. This approach can respond to collaboration requirements change in an "online" manner.

To support the methods mentioned above, this research also adds Protégé container to TH_RTI, changes some services, and realizes the key parts of collaboration ontology based runtime infrastructure, ORTI.

Verified by industrial application, the collaboration ontology based federated development process can keep semantic consistency of collaboration knowledge, improve preparation efficiency of collaborative simulation, and enhance flexibility of HLA federations.

Acknowledgment. This work is supported by Chinese National High-tech Research and Development Program (863 Program, Grant No. 2009AA110302) and Chinese Nature Science Foundation (Grant No. 60874066).

References

1. Simulation Interoperability Standards Committee (SISC) of the IEEE Computer Society, IEEE Standard for Modeling and Simulation (M&S) High Level Archi-tecture (HLA)-Framework and Rules. The Institute of Electrical and Electronics Engineers, Inc., New York (2000)
2. Zhang, H., Wang, H., Chen, D.: Integrating web services technology to HLA-based multidisciplinary collaborative simulation system for complex product development. In: 12th International Conference on Computer Supported Cooperative Work in Design, CSCWD 2008, Xi'an, China, pp. 420–426 (April 2008)
3. Xu, Y., Xiao, T., Liang, C., Zhang, L.: Federated integration of networked man-ufacturing service platforms. Advanced Engineering Informatics 22, 317–327 (2008)
4. Tang, S., Xiao, T., Fan, W.: A collaborative platform for complex product de-sign with an extended HLA integration architecture. Simulation Modelling Practice and Theory 18(8), 1048–1068 (2010)
5. Zhou, W., Liu, Z., Chen, H.: A Survey of the Research about Both FCA and Ontology. Computer Science 33(2), 8–12 (2006)
6. Sun, H., Fan, W., Shen, W., Xiao, T.: Ontology Fusion in HLA-based Collabora-tive Product Development. In: Proceedings of 2010 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2010, Istanbul, Turkey, pp. 2526–2532 (October 2010)
7. Qu, K., Liang, J., Wang, J., Shi, Z.: The algebraic properties of Concept Lattice. Journal of Systems Science and Information 2(2), 271–277 (2004)
8. IEEE Computer Society. IEEE Recommended Practice for High Level Architecture (HLA) Federation Development and Execution Process (FEDEP, IEEE Std 1516.3-2003). The Institute of Electrical and Engineers, NewYork (2003)

PASENS: Parallel Sensor Network Simulator

Banghyun Kim[1] and Jong-Hyun Kim[2]

[1] Ocean Engineering Research Department, Korea Ocean Research & Development Institute, 1312gil 32,Yuseongdaero, Yuseong-gu, Daejeon, 305-343, Republic of Korea
bhkim@moeri.re.kr

[2] Computer and Telecommunication Engineering Division, Yonsei University, Maeji 234, Heungup, Wonju, Kangwon, 220-710, Republic of Korea
Jhkim34@yonsei.ac.kr

Abstract. This paper presents a Parallel Sensor Network Simulator (PASENS) to shorten the time in a large-scale wireless sensor network simulation. The degree of details of the simulation must be high to verify the behavior of the network and to estimate its power consumption and execution time of an application program as accurately as possible. Instruction-level simulation can provide those functions. But, when the degree of details is higher, the simulation time becomes longer. We propose an optimal-synchronous parallel discrete-event simulation method to shorten the simulation time. In this method, sensor nodes are partitioned into subsets, and PCs interconnected through a network are in charge of simulating one of the subsets. Results of experiments using PASENS show, in the case that the number of sensor nodes is large, the speedup tends to approach the square of the number of PCs participating in a simulation. We verified that the simulator provides high speedup and scalability enough to simulate maximum 20,000 sensor nodes.

1 Introduction

Wireless Sensor Network (WSN), which is an infrastructure of ubiquitous computing, consists of a number of sensor nodes of which the hardware is very small and simple. The network topology and routing scheme of the network should be determined according to its purpose. Its hardware and software may have to be changed as needed from time to time. Thus, a WSN simulator, which is capable of verifying its behavior and predicting its performance, is required to improve the design [1].

The WSN Simulator should be able to verify the behavior of WSN, estimate execution time and power consumption, and simulate a large-scale WSN. To satisfy the first two requirements, we use an instruction-level discrete-event simulation (DES), which simulates the behavior of WSN programs at the machine code level with cycle-accuracy. Instruction-level simulation, that is both language and operating system independent, provides the highest behavioral and timing accuracy. But it trades scalability for accuracy. As the degree of details increases, the simulation time becomes longer. Moreover, when the number of sensor nodes increases, the time becomes

J.-H. Kim et al. (Eds.): AsiaSim2011, PICT 4, pp. 15–24, 2012.

extremely long. Thus, we propose an optimal-synchronous Parallel DES (PDES) method to satisfy the last requirement without sacrificing cycle accuracy.

In optimal-synchronous PDES method, sensor nodes are partitioned into a number of subsets, and PCs interconnected through a network are in charge of simulating one of the subsets. The parallel simulation advances with the increment of the Global Virtual Time (GVT) by a time interval between a transmit operation and a receive operation of the RF communication. And each PC performs the simulation with increasing a Local Virtual Time (LVT) asynchronously within a time interval of the GVT.

We have implemented a Parallel Sensor Network Simulator (PASENS) using instruction-level DES and optimal-synchronous PDES. In PASENS, Execution times of most event routines are much shorter than the handling time of an event queue. This means that the handling time is the most significant factor in the simulation time. In this paper we present a speedup formula derived from our simulation model, which shows that the speedup approaches the square of the number of PCs participating in a simulation as the number of sensor nodes increases. The results of our experiments using PASENS prove validity of the formula in predicting the speedup achievable. Instruction traces used as a workload for simulation are executable images produced by the cross-compiler for an Atmega128L microcontroller unit (MCU).

The remainder of this paper is organized as follows: Section 2 reviews related work in PDES. Section 3 describes an instruction-level DES, and Section 4 proposes an optimal-synchronous PDES. Predicting speedup is shown in Section 5. The results of the experimental study with PASENS are reported in Section 6. Finally, Section 7 concludes the paper.

2 Related Work

PDES can solve the problem of slow simulation speed. It is based on partitioning the simulation model into a set of Logical Processes (LPs), which simulate distinct parts of the system under investigation. There are three major PDES techniques; synchronous simulation, conservative simulation, and optimistic simulation. Conservative simulation and optimistic simulation are both asynchronous simulations, where every LP maintains a LVT with generally different clock values at a given point in real time [2].

Synchronous simulation [3] is the simplest simulation technique known. The advantages of synchronous simulation include its low overhead, ease of implementation and performance predictability [4]. However, it is more prone to poor load balancing and high communication cost caused by synchronization steps between cycles. These problems get worse when the number of processors is increased. Synchronous simulation is suited for simulation that has very small computational granularity and reasonably large cycle parallelism [5]. Conservative simulation algorithm dates back to the original works of Chandy and Misra [6] and Bryant [7], and are often referred to as the Chandy-Misra-Bryant (CMB) protocols. Each generated event is accompanied with the LVT of the sending LP. The receiver LP knows that it will not receive any event in future operations with smaller time stamp than the LVT from the sender. Each LP can safely execute the events with a time stamp smaller than the LVT value of all input channels. Time Warp [8] algorithm allows optimistic asynchronous simulation of discrete-event systems. Optimistic mechanisms allow each LP to execute events whenever they are available, thus performing no preventive verification on whether the execution itself meets the correct criterion. On the other hand, if a time stamp order violation is detected, a rollback procedure recovers the LP state to a previous correct value.

GloMoSim [9], QualNet [10], SWAN [11], and SNAP [12] used conservative simulation techniques for WSN simulation. GloMoSim provides a scalable simulation environment for wireless and wired network systems. It is designed using PDES capability provided by Parsec [13]. QualNet is a commercial product of GloMoSim. SWAN is a high performance framework for wireless ad-hoc network simulation, and SNAP achieves conservative simulation using Time-Based Synchronization (TBS) protocol. These simulators can simulate only wireless network simulation, but cannot simulate the activity of each sensor node.

3 Instruction-Level DES

In PASENS, instruction traces used as workloads for simulations are executable images produced by the cross-compiler for an Atmega128L MCU. Intel's hex-record (.hex), Motorola's S-record (.srec), and ROM image (.rom) format are acceptable as workloads. An executable image is loaded onto a Virtual Sensor Node (VSN), which simulates a real sensor board. VSN is an abstract module of the MICAz sensor board including CrossBow MPR2400 [14] and Octacomm NANO-24 [15] . MICAz platform contains an Atmel ATmega128L MCU [16] and a Chipcon CC2420 RF transceiver [17]. Only difference between MPR2400 and NANO-24 is pin assignments.

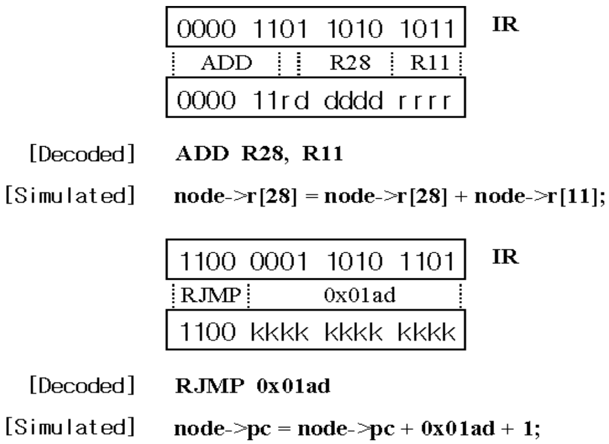

Fig. 1. Examples of instruction simulation