Cleveland State University
EngagedScholarship@CSU

Electrical Engineering & Computer Science Faculty
Publications

Electrical Engineering & Computer Science
Department

1-2005

Design of a Satellite Cluster System in Distributed Simulation

Jong Sik Lee
Inha University, jslee@inha.ac.kr

Thong Luu
Cleveland State University

Vijaya K. Konangi
Cleveland State University, V.KONANGI@csuohio.edu

Follow this and additional works at: https://engagedscholarship.csuohio.edu/enece_facpub

Part of the Systems and Communications Commons

How does access to this work benefit you? Let us know!

Publisher's Statement

Design of a Satellite Cluster System in Distributed Simulation

Jong Sik Lee
School of Computer Science and Engineering
Inha University
Incheon 402-751, South Korea
jslee@inha.ac.kr

Thong Luu
Vijay K. Konangi
Electrical and Computer Engineering Department
Cleveland State University
Cleveland, OH 44115

This article presents the design and development of a satellite cluster system that supports an interfederation communication in High Level Architecture (HLA)–compliant distributed simulation. The interfederation communication enables the execution of a complex, large-scale cluster system of distributed satellites that share the dispersed data assets among satellite components collaboratively. After a brief review of the HLA bridge for interfederation communication, the authors discuss the design issues related to a satellite cluster system that provides cluster management, interfederation communication, and communication data management. They analyze system performance and scalability for centralized and decentralized configurations. The empirical results on the heterogeneous OS distributed system indicate that the satellite cluster system is effective and scalable due to the use of interfederation communication and the reduction of data transmission.

Keywords: Satellite cluster system, interfederation communication, cluster management, communication data management

1. Introduction

A popular trend to execute complex and large-scale systems simulation with reasonable computation and communication resources has been to focus on distributed simulation. High Level Architecture (HLA) [1-3] was designed and has been developed to support the communication among distributed components. HLA was initiated for Department of Defense (DoD) simulations and is a set of specifications that is designed for a distributed simulation. HLA defines the functional elements, interfaces, and design rules needed to achieve a proper interaction of simulations in a federation or among multiple federations. Many researchers have developed and analyzed their model using HLA-compliant simulation environments. A major deficiency of this approach is that most of the HLA-compliant simulations have been achieved with inside-federation communications in only one federation. The simulations in only one federation have shown the limitations when only inside-federation communications among simulation entities are used. In fact, one federation includes multiple federates, and there are multiple federations in an application system. This limitation does not allow communication between two federations. To overcome these limitations and to improve the flexibility in modeling and simulation, this article recommends a distributed simulation environment to allow interfederation communications among multiple federations. Currently, the advantages of HLA bridge interfederation communications in HLA-compliant simulation have been recognized in the literature [4-15] as an important issue for flexibility in modeling and simulation. In this article, we provide the design of a HLA bridge federate to connect multiple federations in a satellite cluster system. This bridge-based interfederation communication can contribute to modeling and simulation flexibility and thus enable us to construct a complex and large-scale distributed system and analyze its performance using simulation-based empirical data.

Recently, the trend in satellite communications has been to use constellations of autonomous spacecraft (satellites) that can function collaboratively. Many mission-related organizations, including the National Aeronautics and Space Administration (NASA) and DoD, are conducting research to use these constellations to accomplish their mission

objectives. The principal goals of constellations are to have autonomous control over data collection, autonomously perform scientific analysis of large data sets such as radar images, and return the data with the highest scientific value as well as selected scientific analyses of the data to the ground stations, thus negating the need to return the entire data set. For an effective execution of this constellations system, a cluster paradigm [16, 17] has been noticed. This article proposes the high-performance modeling and simulation for this autonomous constellations system with distributed system construction and execution concepts. In particular, the bridge-based interfederation communication in HLA-compliant distributed simulation supports the flexible and high-performance modeling for not only communication among satellites within the cluster but also communication between two clusters and between the cluster and ground system. This article focuses on an interfederation communication system design when a cluster manager is assigned in a bridge federate. In addition, this article proposes a communication data management method with centralized and decentralized approaches to improve the satellite cluster system performance. The methodology is extended from communication data management [18-25] to execute a complex and large-scale distributed system with reasonable computation and communication resources. In our methodology, we provide a functionality balancing for each satellite inside the cluster and the intelligence of the satellite. *Functionality balancing* refers to the distribution of functions to each satellite from a centralized satellite or a ground system, and this balancing reduces the total system execution cost through satellite local computation load balancing. Satellite intelligence corresponds to each satellite operating autonomously and making its own decisions, in addition to its basic operations such as image capturing, data transferring, and so on. This method increases system modeling flexibility and improves system performance through a reduction of communication data and synchronization of computation, as well as advances system scalability for a large-scale constellations system.

This article is organized as follows: section 2 introduces satellite cluster management using a central manager and discusses the role of the manager and a federate assignment/recovery in a cluster federation. Section 3 introduces an HLA bridge for connection among multiple federations and describes a design of a satellite cluster system with interfederation communication and practical implementations. Section 4 presents the communication data management between a cluster and a ground system and analyzes performance effectiveness and system scalability. Section 5 introduces a testbed for the experiment, evaluates the performance of centralized and decentralized data management, and discusses interfederation and inside-federation communications. The conclusions are in section 6.

2. Satellite Cluster Management

The construction and execution of an autonomous constellations system follows distributed system construction con-

cepts: (1) functionality balancing in multiple distributed satellites, (2) increasing system robustness and maintainability, and (3) reduction of communication and computation resources. Distributed satellite functionality includes command and control, communications, and payload functions. For effective execution of a constellations system, a cluster paradigm with a central cluster manager is modeled and simulated in this article. The central cluster manager controls the functionality of each satellite inside the cluster and communication among satellites.

Separated spacecraft in a cluster occupy their distributed space assets in a constellation system. The cluster management of the distributed assets is essential for a satellite cluster mission to accomplish cluster functionalities such as resource management, navigation, guidance, fault protection, and so on. While a centralized management approach is defined, a cluster manager provides the cluster functionalities. The operation of a cluster manager consists of four categories: spacecraft command and control, cluster data management, flying formation, and fault management. For cluster data management, a cluster manager should keep the information of each spacecraft, including position, velocity, attitude quaternion, system time, spacecraft mode, fuel level, sensor states, and so on. A very common problem of centralized data management with a cluster manager is the recovery of satellite system operations when the cluster manager is nonfunctional. Here, we introduce a recovery algorithm in a two-federation system on an HLA/RTI (runtime infrastructure) platform, including cluster and ground federations. In the basic HLA design, an HLA-compliant system architecture is specified as component integration for cooperation in distributed simulation. The design of architecture provides an independently developed simulation, which is called a federate, and a set of simulations, which is called a federation. RTI coordinates the interactions among the simulations of a federation and performs a basic mechanism for initializing, directing, and controlling the flow of data exchange among simulations. This article specifies the recovery algorithm of a satellite federate in a cluster federation as the following:

- Attribute Definition

    - ManagerNumber (0,1,2,3, or MAX_SAT (4)) (If a satellite is the manager, it sets its ManagerNumber as its satellite number.)
    - AmIManager (TRUE, FALSE)

- When a satellite launches, it checks if the cluster federation is present.

    - If no, the satellite should be the manager (sets AmIManager as true and ManagerNumber as its satellite number).
    - If yes, the satellite sets AmIManager as false and sets its ManagerNumber to MAX_SAT.

- Satellite class update routine.

– If the satellite is the manager, nothing needs to be done.
– If the satellite is not the manager, check the ManagerNumber of all other satellites.
– If any of the other satellites is 0-3, the satellite sets its ManagerNumber to MAX_SAT.
– If any of the other satellites are SAT_MAX, the new manager satellite should be the one that has the smallest satellite number.
– The new satellite sets its ManagerNumber to its satellite number and sets AmIManager to true.

• When a satellite exits, it exits specified federations.

– If the satellite is the cluster manager, it needs to exit from both the cluster and ground federations.
– If it is not the cluster manager, it just exits the cluster federation.

3. Satellite Cluster System with Interfederation Communication

3.1 Usefulness of the HLA Bridge in Satellite Cluster System Modeling and Simulation

An HLA-compliant system architecture is specified as a component integration for cooperation in distributed simulation. A federation is composed of a single RTI coordinating a single set of federates. In practice, a federate is associated with an application entity and a federation as a group of entities in most simulation applications. In the research presented in this article, we are interested in developing a composite federation to make individual federations work together. The composite federation satisfies the basic objective of distributed simulation, which is to improve the interoperability among separated simulation entities, even though it has displayed restrictions for communication across federations, and one federation can only get information transferred from another federation. Here, we discuss the hierarchy of an HLA-compliant system. A federation includes multiple federates, and an application system includes multiple federations. This article provides an integration among components of a satellite cluster system and constructs the larger mission system by suggesting an HLA bridge connection among multiple federations. The integration among components should support construction flexibility of a satellite cluster system, and thus the integration allows us to reduce system structure limitations when a satellite cluster system is developed. In addition, the integration provides system scalability to make and execute a large-scale satellite cluster distributed system.

To make a connection among multiple federations, the bridge federate is used. The bridge federate is a common federate of the federations, which are associated with taking part in communication to specified federations. The bridge federate is physically located in each federation
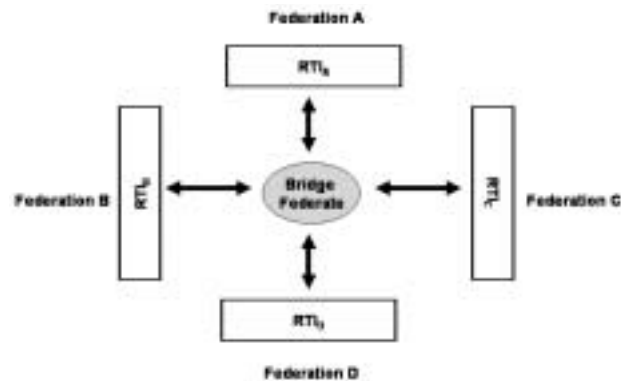

Figure 1. Bridge federate connected to multiple federations

and routes RTI messages to proper destination federations, as shown in Figure 1. The bridge federate would provide specified and useful services, including message filtering/distribution and data translation/generation.

3.2 Design of a Satellite Cluster System with Interfederation Communication

In this section, we discuss an interfederation communication architecture on a multifederation platform and apply the interfederation communication architecture to the data management of a satellite cluster system. In contrast to conventional HLA architectures, which use inside-federation communication within one federation, as Figure 2 illustrates, we apply inside-federation communication to the data management of a satellite cluster system and develop a cluster system federation. The satellite cluster system includes a cluster of satellites and a ground station. The satellite cluster is composed of four spacecraft, with one spacecraft serving as the cluster manager. There are total of five simulation entities, and we assign each simulation entity to a federate inside a federation.

Since we would like to extend our design methodology to larger satellite cluster systems, we also propose an interfederation communication architecture. In general, a satellite cluster system can be separated into two divisions: the space cluster and the ground station. This separation enables us to differentiate between the simulation entities for reasons of geography, functionality, and being members of different communication groups. For example, typically, a ground station has connections to various organizations on Earth, and thus it is a part of a different communication group. We apply the two divisions (e.g., space cluster and ground station) to an advanced HLA architecture, which is the interfederation communication architecture. To execute the interfederation communication, we use the bridge federate, which is physically located in each federation and is responsible for passing RTI messages between federations.
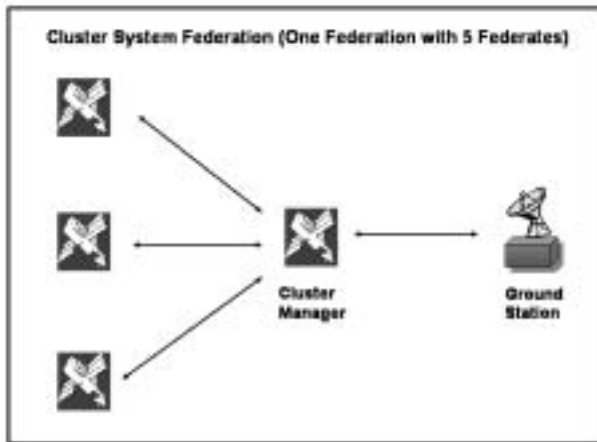
Figure 2. Inside-federation communication in a satellite cluster system
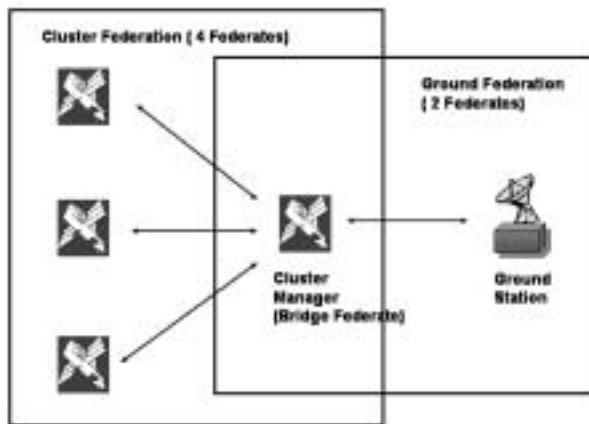

Figure 3. Interfederation communication in a satellite cluster system

As Figure 3 illustrates, we develop two federations: cluster and ground. The cluster federation includes four federates, and each spacecraft inside the cluster is assigned to a federate. The ground federation includes two federates: cluster manager and ground station. Both federations include the cluster manager federate, which is assigned in a bridge federate for the interfederation communication. Notice that the cluster manager federates in both federations have appropriately different functionalities. The cluster manager federate in the cluster federation is tasked with cluster data management by passing RTI messages inside the federation and for interfederation communication.. The cluster manager federate in the ground federation only concentrates on communication with RTI message passing to the cluster federation.

3.2.1 Fed File Implementation

To execute interfederation communication in the satellite cluster system, we develop two federations, cluster and ground, as noted previously. Each federation needs its fed file. The fed file specifies RTI communication, including interaction or attribute communications, and assigns the communicated attributes as the following:
Cluster.fed

- Class satellite:
    - Attributes: name, satellite number, ManagerNumber, position (x, y, z) (latitude, longitude, altitude)
    - Interactions: communication (message)

Ground.fed

- Classes: ground station and cluster manager
    - Class cluster manager:
        * Attributes: name, satellite number, position (x, y, z)
    - Class ground station:
        * Attributes: name, position (x, y, z)
        * Interactions: communication (message)

4. Data Management in a Satellite Cluster System

In this section, we discuss scalability issues of a satellite cluster system. Here, we introduce a ground system operation as a case study to discuss centralized and decentralized approaches to data management, and we then evaluate performance of each approach. Customarily, a ground system commands and controls a cluster of spacecraft. The ground system requires operations to monitor the cluster, make decisions, formulate proper command strings, and transmit the command strings to the cluster. For a small cluster, a centralized approach is cost effective and expected to command and control the spacecraft individually. As Figure 4 illustrates, a ground system sends the command strings to each spacecraft. The command strings include a command to "observe a specified region, take a picture, and send the image data." The command should contain the region location. Each spacecraft receives different region location from the ground station.

To optimize the ground operation cost, a decentralized approach for ground operations is proposed in this article. The decentralized approach indicates that it separates ground functions and distributes a set of functions to spacecraft. The ground station separates the four regions to be observed, makes four different command strings, and sends them to a cluster manager. The cluster manager parses the command strings and forwards them to the proper spacecraft. Figure 5 illustrates a decentralized approach in satellite cluster system operation. Here, we introduce the concept of decentrality. Decentrality refers to the distribution
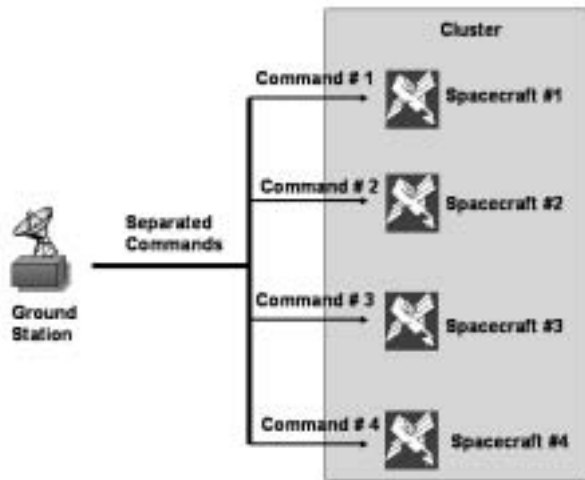
Figure 4. A centralized approach in satellite cluster system operation

Figure 5. Decentralized approach in satellite cluster system operation

of functions or operation loads to each distributed component. Most of the current research reported in the literature related to load balancing focuses on execution time reduction through concurrent processing with load balancing. Our approach, however, focuses on execution time reduction through communication-required data reduction with load balancing. Of course, concurrent processing with load balancing helps in the reduction of execution time, although the effect is marginal. In this case study, we investigate two degrees of decentrality: low and high. In the decentralized low-intelligence case, the parsing and forwarding of ground station command strings to the appropriate satellite is assigned to a cluster manager. The parsing and forwarding classifies a lower intelligence of a cluster manager. In the case of high decentrality, the ground station does not separate the four regions to be observed but sends the total region to be observed to a cluster manager. The cluster manager should include the intelligence for division of the region to be observed. The division intelligence should understand the technology, including region division, image capturing, image visualization, image data aggregation, data transmission, and so on.

4.1 Analysis of Transmission Data

To evaluate the system performance of the centralized and decentralized approaches, we compare the amount of transmitted data between the ground station and the satellite cluster. Note that transmitted data among spacecraft within the cluster are ignored. In this analysis, we assume the following conditions to simplify the analysis: (1) there exist multiple clusters communicating to the same ground station;(2) a cluster includes a fixed number of spacecraft
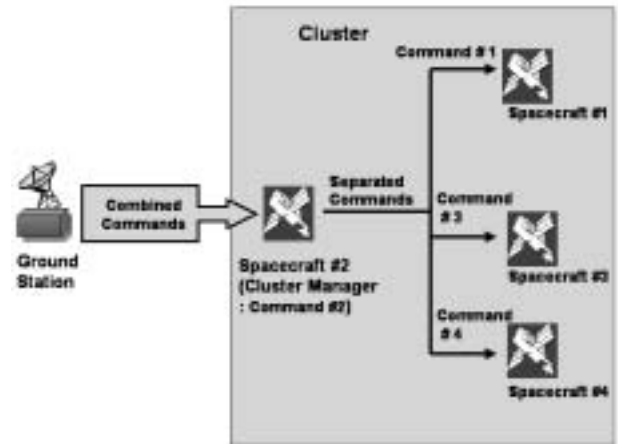
(N); (3) the region is square shaped, and there are four points ((x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)) to identify the region; (4) each point is represented by a pair of double-precision-type variables (x, y), and each double-precision variable is represented by 32 bits (i.e., each point requires 64 bits); and (5) the analysis is based on one cycle of transmission.

As Table 1 shows, the decentralized approach significantly reduces the number of required communication messages and the number of required communication bits. Basically, overhead bits (H) are needed for satellite communication when a ground station sends a command string. The centralized approach causes an amount of overhead messages and corresponding bits to be transmitted since the ground station has to transmit messages to the spacecraft individually. Comparing the two degrees of decentrality in the decentralized approach, the high decentrality significantly reduced the required communication data bits since it transmits the location information for one big region, and the number of spacecraft (N) in a cluster is irrelevant. In particular, as the number of spacecraft (N) tends toward infinity, the required communicated data bits in the low-decentrality case increase linearly. The slope of the increase is $(4 \cdot 64) \cdot M$. However, the high decentrality still requires the same or lower number of required communication data bits. The analysis in Table 1 reveals that especially when a large number of spacecraft are working in a cluster, we expect the largest reduction in required communication data in the decentralized approach with high decentrality. In the decentralized approach with high decentrality, there is computation overhead to execute the operations of a cluster manager. However, we can ignore this overhead since the communication resource is a critical factor to execute a satellite cluster system within reasonable time.

Table 1. Analysis of communication-required data reduction

Approach		Number of Required-Communicated Messages	Number of Required-Communicated Messages Bits	Coefficient of of N as $N\rightarrow\infty$	Coefficient of N & M as $N\rightarrow\infty$ & $M\rightarrow\infty$
Centralized		$R \cdot N \cdot M$	$(H + 4 \cdot 64) \cdot R \cdot N \cdot M$	$(H + 4 \cdot 64) \cdot R \cdot M$	$(H + 4 \cdot 64) \cdot R$
Decentralized	Low	M	$(H + 4 \cdot 64 \cdot R \cdot N) \cdot M$	$(4 \cdot 64 \cdot R) \cdot M$	$(4 \cdot 64 \cdot R)$
(Decentrality)	High	M	$(H + 4 \cdot 64) \cdot M$	None	None

N = number of spacecraft in a cluster; M = number of clusters; H = number of overhead bits in satellite communication (160 bits assumed); R = number of regions at one spacecraft on one transmission (40 bits assumed).

5. Experiment and Results

5.1 Case Study and Testbed

To evaluate the performance of the satellite cluster system simulation, we introduce a scenario of satellite cluster management.

5.1.1 Scenario

A cluster of four spacecraft flies on prescheduled orbits. One of the spacecraft acts as a cluster manager that communicates with a ground station. For data communication on RTI, SendInteraction() and ReceiveInteraction() functions of RTI API are used. The cluster manager gathers the states of each spacecraft and sends telemetry information back to the ground station. At any given time, the ground station can send an observation request to the cluster manager, which in turn will coordinate with other spacecraft in the cluster to perform the requested observation in synchronization. The cluster manager then aggregates data collected from the other spacecraft and sends them back to the ground station.

In our experiments, the ground station initiates the sequence for communication and computation between the ground station and the satellite cluster by transmitting a request. We collected empirical data by sending one request from the ground station that the experimenter makes by using a button on the graphical user interface. The computations performed by an individual satellite depend on the role of that satellite in the cluster. Upon receiving a message from the ground station, the cluster manager will use the received message to formulate messages to send to the other satellites in the cluster. In the decentralized high-intelligence case, the cluster manager will need to compute the location data for each satellite member. In the decentralized low-intelligence case, the cluster manager just decomposes the message and transmits to the members of the cluster. Upon receiving a request message, a member satellite will process the request by simulating observation. In a real satellite system, the cluster manager would need to aggregate the image data received from the satellites in the cluster. But in our experiment, no aggregation of data is done; the manager simply waits to receive all acknowledgments transmitted back by the members of the cluster and then sends one acknowledgment to the ground station. The messages are modeled as an Interaction (with parameter) in HLA/RTI terms. Most of data are sent using Interaction (with parameters), and only health data are sent using Object (with attributes) with functions of updateAttributeValues() and reflectAttributeValues(). In our experimental setup, we have no control on the packet size that is used. This is because HLA/RTI works in the application layer; all communication is done by RTI, and there is no control over the lower layers.

5.1.2 Assumptions

- The cluster manager always communicates with the ground station without interruption.
- The position representation of each spacecraft is relative to the reference circular orbit.
- All spacecraft fly at an altitude of 600 km on the reference circular orbit. This yields a period of 5810 seconds for each orbit.

To execute the scenario, we developed two testbeds for inside-federation and interfederation communications. As Figure 6 illustrates, the inside-federation communication works on a cluster/ground federation. The federation includes varying numbers of spacecraft federates, including a cluster manager, and one ground station federate. The RTI message passing for cluster data management depends on the inside-federation communication. In our testbed, the RTI implementation was based on RTI-1.3NGv6. We developed a heterogeneous distributed system that included various operating systems composed of Windows 2000, Linux, and SGI IRIX. For the basic configuration used in this research, CPUs were the Compaq Pentium, Dell PC, and SGI Octane2. Two Windows machines were used for the cluster manager federate and ground station federate. For the three federates for three spacecraft, the operating systems and corresponding CPUs were IRIX on SGI Octane2 and Linux on the Dell PC. All machines were connected via a 10 Base T Ethernet network.

For inter-federation communication with a bridge federate, as Figure 7 illustrates, the cluster federation includes four spacecraft federates, including a cluster manager, and the ground federation includes two federates: cluster manager and ground station. Both federations have the cluster
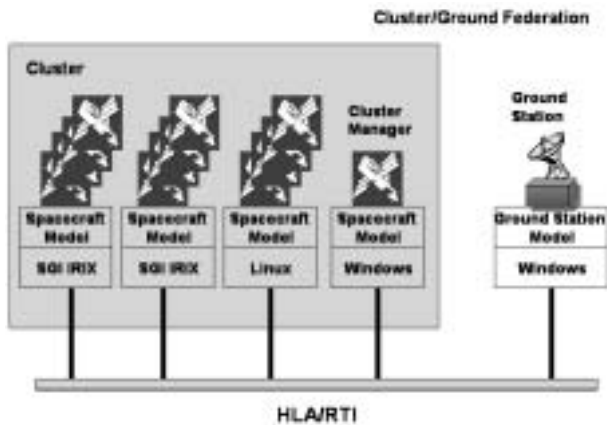
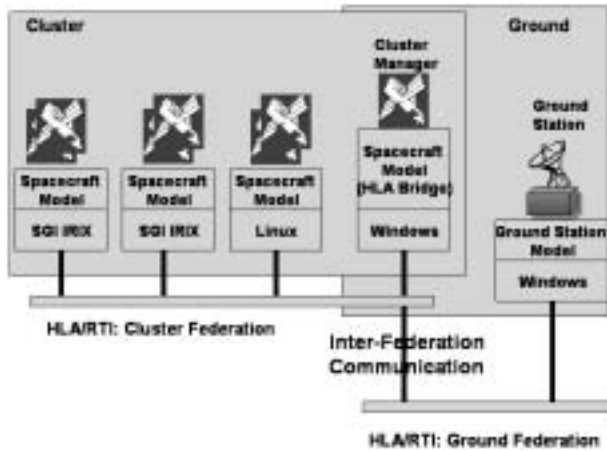**Figure 6.** Inside-federation communication in a cluster/ground federation

Figure 7. Inter-federation communication between cluster and ground federations

manager federate, which is called the bridge federate. In our model, we employed a package for HLA Bridge operation on the spacecraft model layer. The package includes a set of functions that perform interfederation communications. The functions are used only for the cluster manager federate, although the package exists in the implementations of all federates.

5.2 Effect of Centralized and Decentralized Data Management in a Satellite Cluster System

To evaluate system execution performance of the centralized and decentralized data management, we compare data bits transmitted in the following cases: centralized, decentralized with low decentrality, and decentralized with high
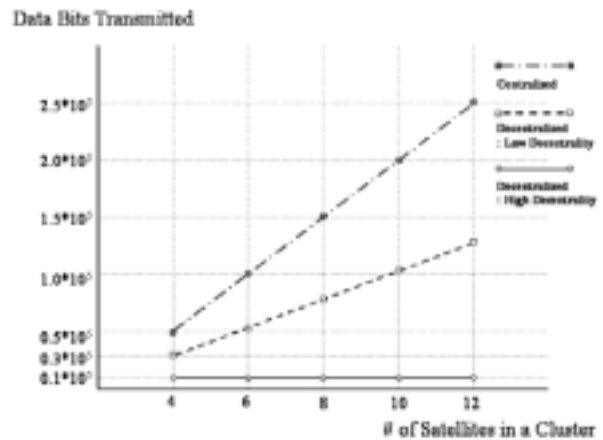

Figure 8. Comparison of data bits transmitted in centralized and decentralized approaches

decentrality. The evaluation is performed by varying the number of satellites in a cluster. The centralized approach is executed on only one federation, which provides inside-federation communication as shown in Figure 6. The decentralized approach is executed on two federations: cluster and ground. The execution of the two federations provides inter-federation communication, as shown in Figure 7.

As Figure 8 illustrates, the decentralized approach, for both the high and low degrees of decentrality, results in a considerable reduction of the data bits transmitted. In addition, in the decentralized approach with high decentrality, there is a very significant reduction in the data bits transmitted. Furthermore, in the latter case, the results indicate that the execution requires a small amount of transmitted data bits and is independent of the number of satellites in the cluster.

The second measure to evaluate system execution performance is system execution time. The system execution time consists of both communication and computation performance. The centralized approach requires a large amount of communication data bits. However, it reduces the local computation of each satellite. The system execution time for the centralized approach is mostly caused by the amount of communication data. The decentralized approach, on the other hand, reduces the amount of communication data and increases the computation load of the cluster manager. The system execution time for the decentralized approach is caused by both data communication time and local computation time. The high-decentrality case especially requires more local computation time of the cluster manager than that for low decentrality. Figure 9 compares system execution time in the centralized and decentralized approaches. The system execution time of Figure 9 is obtained from the execution on only one federation
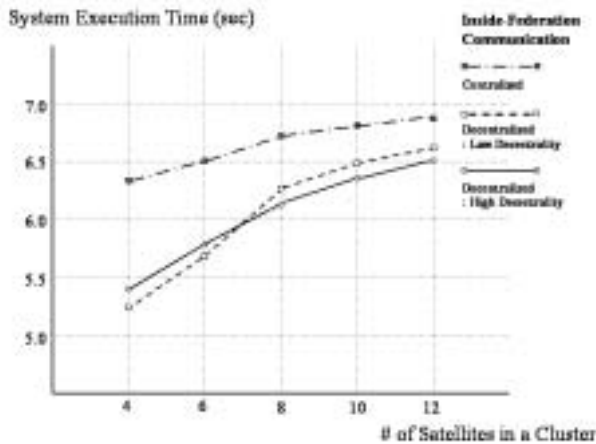
Figure 9. System execution time of centralized and decentralized approaches on inside-federation communication

Figure 10. System execution time of centralized and decentralized approaches on interfederation communication

with inside-federation communication. The decentralized approach reduces the system execution time, as shown in the figure. The system execution time reduction indicates that time reduction from communication is greater than time expense from local computation time. Here, we can find that there exists a trade-off between transmission data reduction and local computation time in the comparison between high and low decentrality. In inside-federation communication system of Figure 9, the low decentrality performs the lower execution time in the lower task load. The smaller number of satellites presents the lower task load. As the task load increases, the high decentrality requires the lower execution time. Figure 10 illustrates the system execution time with interfederation communication on the two federations. The results for the centralized approach are not reported since this approach cannot be operated with interfederation communication. The high decentrality requires the lower execution time for the lower task load. As the task load increases, the low decentrality increases its execution time slowly, and its execution time is comparable to that of high decentrality.

5.3 Discussion of Interfederation and Inside-Federation Communications in a Satellite Cluster System

To evaluate system execution performance of an interfederation communication system, we compare system execution time of interfederation communication with that of inside-federation communication with only one federation. As we mentioned in section 4, an interfederation communication system is operated with a bridge federate between the two federations: cluster and ground. In general, an interfederation communication system reduces its local computation time. An interfederation communication system
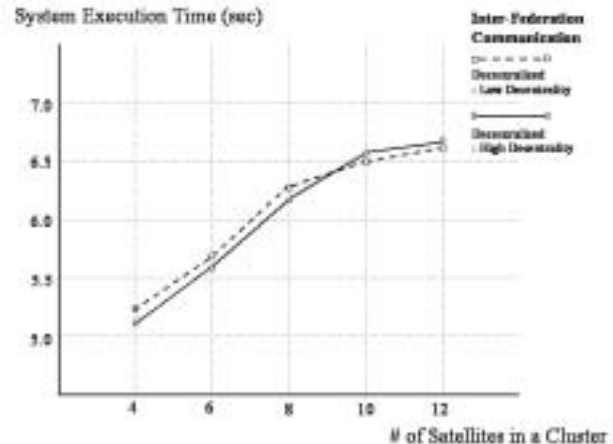
includes multiple federations; thus, it separates its tasks and assigns the subtasks on each federation. However, an interfederation communication system increases its communication time since interfederation message-passing time would be greater than inside-federation message-passing time in a federation. Also, we can expect that the inside-federation communication system needs high local computation time and low communication time.

Figures 11 and 12 compare system execution time of the two communications: inside federation and interfederation. We measure system execution time in the case of both low and high decentrality of the decentralized approach. System execution time of the inside-federation communication system is lower for the low task load (e.g., small number of satellites) with low decentrality, as shown in Figure 11. As the task load increases, the execution time of the interfederation communication system is lower. Figure 12 shows system execution time with high decentrality. The execution time of the interfederation communication system is lower in all the tasks, but only marginally. For higher task loads, the execution times of the two communication systems are approximately the same. Finally, we noticed a trade-off between communication time and local computation time in interfederation and inside-federation communication systems.

6. Conclusions

This article presented the design and development of a satellite cluster system taking advantage of distributed simulation. For practical construction and execution of an autonomous constellation system, we employed distributed system construction concepts, including data communication management, HLA-compliant interfederation communication, functionality balancing, system model-
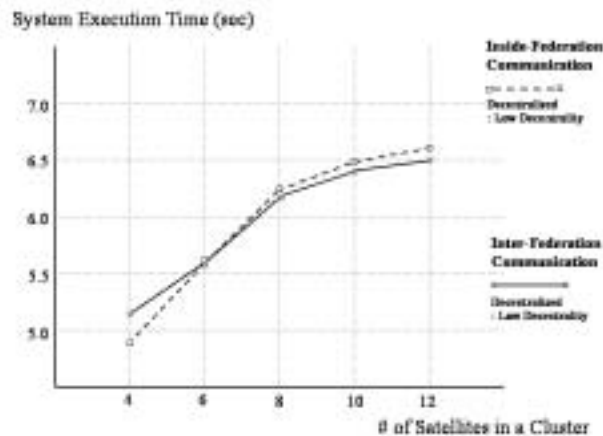
Figure 11. System execution time of interfederation and inside-federation communication with low decentrality
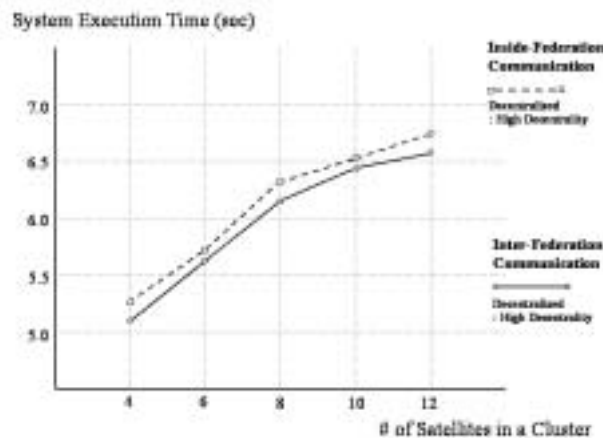

Figure 12. System execution time of interfederation and inside-federation communication with high decentrality

ing flexibility, scalability, and maintainability, and provided high-performance modeling for the cluster paradigm with a cluster manager. As noted in this article, bridge-based interfederation communication in HLA-compliant distributed simulation improved the modeling flexibility and scalability by allowing multiple connections not only among satellites inside the cluster but also among clusters and between the cluster and ground systems. This modeling flexibility allowed us to model and simulate topologies of a variety of autonomous constellation systems and thus analyze a complex, large-scale space mission system and obtain empirical results. In particular, our results pointed to a trade-off between communication time and local computation time in interfederation and inside-federation com-

munication systems. On the basis of these results, we would recommend the following for system modeling and simulation: (1) interfederation communication would be recommended when the focus is on flexibility and scalability of system modeling and simulation, (2) interfederation communication would be recommended when communication time is a critical factor of system simulation performance, and (3) inside-federation communication would be recommended when local computation time is a critical factor of system simulation performance.

This article provided a data communication management method with centralized and decentralized configurations. The method focuses on functionality balancing in each satellite and various degrees of functionality decentrality in each satellite, including the cluster manager. The functionality decentrality permits function distribution to each satellite from the centralized functions of a conventional ground system. In addition, the methodology indicates an active conversion from a conventional passive satellite, which processes given commands, to an intelligent satellite, which processes its own decisions. This decentrality of functionality allows modeling and simulation for a variety of autonomous constellation systems and improves system performance through data communication reduction and computation load balancing. We analyzed system performance and scalability of the centralized and decentralized approaches for data management. The empirical results showed favorable reduction of communication data and overall simulation time and demonstrated the usefulness of the satellite cluster system in scalable distributed modeling and simulation. In our future research, we plan to extend the satellite cluster system to a real-time satellite cluster distributed simulation and execution infrastructure. Bridge-based interfederation communication in HLA-compliant distributed simulation and communication data management can enable us to overcome the constraints of real-time simulations.

7. Acknowledgments

8. References

[1] *High Level Architecture run-time infrastructure programmer's guide 1.3, version 3*. 1998. Alexandria, VA: DMSO.

[2] Kuijpers, Nico, Johan Lukkien, Bas Huijbrechts, and Marco Brassé. 1999. Applying data distribution management and ownership management services of the HLA interface specification. In *Proceedings of the Fall Simulation Interoperability Workshop*, 99F-SIW-023, Orlando, FL.

[3] *Draft standard for modeling and simulation (M&S) High Level Architecture (HLA)—Federate interface specification* [draft 1]. 1998. Alexandria, VA: DMSO.

[4] Filsinger, J. 1996. HLA secure combined federation architecture (part one). Technical report, Trusted Information Systems.

[5] Braudaway, W., and R. Little. 1997. The High Level Architecture's federation bridge. In *Proceedings of the Fall Simulation Interoperability Workshop*, 97F-SIW-078, September.

[6] Lake, T. 1998. Time management over inter-federation bridges. In *Proceedings of the Fall Simulation Interoperability Workshop*, 98F-SIW-099, September.

[7] Gan, Boon Ping, Dan Chen, Nirupam Julka, Stephen J. Turner, Wentong Cai, and Guangya Li. 2003. A comparison of alternative topologies for multi-level federations. In *Proceedings of the 2003 Spring Simulation Interoperability Workshop*, 03S-SIW-027, March, Orlando, FL.

[8] Myjak, M., R. Carter, D. Wood, and M. Petty. 1998. A taxonomy of multiple federation executions. In *Proceedings of the 20th Interservice/Industry Training Systems and Education Conference*, November, pp. 179-89.

[9] Dingel, J., D. Garlan, and C. Damon. 2001. A feasibility study of the HLA bridge. Tech. Rep. CMU-CS-01-103, Department of Computer Science, Carnegie Mellon University.

[10] Cramp, A., and M. Oudshoorn. 2002. Employing hierarchical federation communities in the virtual ship architecture. In *Proceedings of the Twenty-Fifth Australasian Conference on Computer Science*, vol. 4, pp. 41-9.

[11] Dingel, J., D. Garlan, and C. Damon. 2002. Bridging the HLA: Problems and solutions. In *Proceedings of the Sixth IEEE International Workshop on Distributed Simulation and Real-Time Applications*, October, pp. 33-42.

[12] Cai, Wentong, Stephen J. Turner, and Boon Ping Gan. 2001. Hierarchical federations: An architecture for information hiding. In *Proceedings of the 15th Workshop on Parallel and Distributed Simulation*, pp. 67-74.

[13] Bouwens, C., D. Hurrell, and D. Shen. 1998. Implementing ownership management with a bridge federate. In *Proceedings of the 1998 Spring Simulation Interoperability Workshop*, 98S-SIW-197, March, Orlando, FL, pp. 1126-31.

[14] Lake, Tom, Colin Rayment, and Kwok Wong. 2000. A component framework for hierarchical simulation. In *Proceedings of the 2000 Fall Simulation Interoperability Workshop*, 00F-SIW-083, September, Orlando, FL.

[15] Magee, Gerry, Graham Shanks, and Pete Hoare. 1999. Hierarchical federations. In *Proceedings of the 1999 Spring Simulation Interoperability Workshop*, 99F-SIW-085, March, Orlando, FL.

[16] Zetocha, P. 2000. Intelligent agent architecture for onboard executive satellite control. In *Intelligent automation and control*, vol. 9, 27-32. Albuquerque, NM: TSI Press.

[17] Surka, D. M., M. C. Brito, and C. G. Harvey. 2001. The real-time ObjectAgent software architecture for distributed satellite systems. In *Proceedings of the IEEE Aerospace Conference*, vol. 6, pp. 2731-41.

[18] Zeigler, Bernard P., Hyup J. Cho, Jeong G. Kim, Hessam Sarjoughian, and Jong S. Lee. 2002. Quantization based filtering in distributed simulation: Experiments and analysis. *Journal of Parallel and Distributed Computing* 62 (11): 1629-47.

[19] Morse, Katherine L. 1996. Interest management in large scale distributed simulations. Tech. Rep. 96-127, Department of Information and Computer Science, University of California, Irvine.

[20] Morse, Katherine L., Lubomir Bic, Michael Dillencourt, and Kevin Tsai. 1999. Multicast grouping for dynamic data distribution management. In *Proceedings of the 31st Society for Computer Simulation Conference*.

[21] Saville, J. 1997. Interest management: Dynamic group multicasting using mobile Java policies. In *Proceedings of the 1997 Fall Simulation Interoperability Workshop*, 97F-SIW-020.

[22] Boukerche, A., A. Roy, and N. Thomas. 2000. Dynamic Grid-based multicast group assignment in data distribution management. In *Proceedings of the 4th IEEE International Workshop on Distributed Simulation and Real-Time Applications*, pp. 47-54.

[23] Tan, G., Y. Zhang, and R. Ayani. 2000. A hybrid approach to data distribution management. In *Proceedings of the 4th IEEE International Workshop on Distributed Simulation and Real-Time Applications*, August, pp. 55-61.

[24] Lee, Jong S., and Bernard P. Zeigler. 2002. Space-based communication data management in scalable distributed simulation. *Journal of Parallel and Distributed Computing* 62:336-65.

[25] Lee, Jong Sik, and Bernard P. Zeigler. 2002. Design and development of data distribution management environment. *SIMULATION* 77:39-52.