Design and Implementations of Surrogates for Interoperation of HLA Federations

Min Wook Yoo and Tag Gon Kim
School of EECS KAIST
373-1 Kusong-dong, Yusong-gu
Daejeon, Korea 305-701
Tel: +82-42-869-5454
Fax: +82-42-869-8054
mwyu@smslab.kaist.ac.kr, tkim@ee.kaist.ac.kr

KEYWORDS
HLA, RTI, INTEROPERATION, FEDERATION, SURROGATE, BRIDGE

ABSTRACT: HLA (High Level Architecture) is a specification for interoperation among distributed heterogeneous simulators which are developed in different languages and platforms. HLA originally allows a number of federates to join in a federation using a single Run-Time Infrastructure (RTI). HLA does not specify the standard for interoperation of federations and RTI does not supply the related services. Therefore, there are several problems for interoperation of federations. This paper proposes a new architecture for interoperation of a set of federations, which employs a 2-way Surrogate based on an HLA Bridge. The architecture supports interoperation of two federations. As a representative of one federation, a Surrogate joins the opposite federation, which communicates with another Surrogate and federates of its joined federation. To keep HLA Rules, Surrogate and federates in the same federation do not exchange FOM data.

1. Introduction

HLA (High Level Architecture) is the standard for interoperation of distributed simulators and RTI (Run-Time Infrastructure) is the software which supplies services defined in HLA. Individual simulators called federates should organize a federation using a RTI for interoperation. A federation shall have a FOM (Federation Object Model) and a federate shall have a SOM (Simulation Object Model) [1]. FOM includes all objects and interactions which can be exchanged in the federation [2]. All federates in a federation can access the data recorded in its FOM. SOM includes the data which a federate provides to a federation and receives from a federation.

A federation means the integrated simulation of federates. There are some cases in which integration of federations is needed. For example, there are two federations which are developed by different developers and they want to combine the federations to perform an integrated simulation. Figure 1 describes interfaces specified in HLA. HLA specifies the standard for interoperation of federates, but not for interoperation of federations. Therefore, current RTIs do not supply the services for interoperation of federations.

Integration of multiple federations can be made in two methods. One is creating a new federation in which all federates in those multiple federations are joined. In this case, information recorded in FOM is accessible to all federates which join the federation. This may cause a serious security problem. Because RTI cannot hide FOM from federates of a federation, this problem cannot be solved. The other is interoperating the existing federations using a mediating unit. In this case, security problems can be solved but there still exists remaining problems such as consensus. Therefore, these problems should be solved for successful interoperation.

A simple solution is modifying RTI to support interoperation of federations [3][4]. In this case, existing RTI cannot be used and available RTIs are limited.
There has been research into the use of the mediating unit for interoperation of federations [4][5][6]. These approaches also have problems. Several problems can be solved by rigorous implementation. However, problems which arise due to unobtainable information cannot be solved in existing architecture. These problems can be solved by extension of RTI interfaces [7][8] but then RTI should be modified.

To solve those problems without modifying RTI, this paper proposes a new architecture for interoperation of federations. Proposed 2-way Surrogate uses a Surrogate as a representative of one federation. A Surrogate joins the opposite federation to represent behavior of the federation. Like HLA Bridge, Surrogate acquires the federation information from RTI. In addition, Surrogate uses direct communication paths from federates. To keep HLA rules, FOM data is not sent through direct paths.

Section 2 introduces HLA Bridge which is base structure of 2-way Surrogate and its problems. Section 3 describes design of 2-way Surrogates. Implementation of 2-way Surrogate is explained in Section 4. Section 5 shows the experimental results. Section 6 concludes the paper.

2. Background

2.1. HLA Bridge [5]

HLA Bridge is the architecture for interoperation of federations without modifying RTI. For interoperation, HLA Bridge uses surrogates which represent the behavior of federations.

2.1.1. Architecture

Figure 2 shows the architecture of HLA Bridge. HLA Bridge consists of two surrogates and a transformation manager.

![Figure 2 HLA Bridge [5]]

Surrogate $S_F$ is a federate representing behavior of federation $F$. $S_F$ requests services in federation $G$ by using the information from $S_G$. Also $S_F$ sends information of federation $G$ to $S_G$. Surrogate $S_G$ is a federate representing behavior of federation $G$. Transformation Manager $TM$ is a module which translates the two FOMs of federations for mapping the entities.

2.2. Problems of HLA Bridge

Because bridge is not defined in HLA, RTI treats a surrogate as a normal federate. Therefore, surrogates communicate with RTI by only HLA services. When a bridge needs the information which cannot be acquired by using HLA services, some problems occur. Federate failure is the problem that RTI does not allow the failure of the service. Consensus is the problem that a bridge cannot know the service completion of other federates and the service is not completed. Service barrier is the problem that service barriers do not disappear simultaneously in federations. Insufficient information is the problem that a bridge cannot acquire sufficient information to represent a federation. Selective addressing is the problem that a partial synchronization set cannot contain federates of opposite federation.

Problems can be divided into two categories. One is the problem which a bridge recognizes and can solve by itself. Federate failure and service barrier belong to this category. The other is the problem which a bridge cannot solve by itself. Consensus and insufficient Information belong to this category. Problems of later category cannot be solved in current RTIs.

3. Design of 2-way Surrogate

This paper proposes the 2-way Surrogate for interoperation of two federations. Like HLA Bridge, 2-way Surrogate uses a Surrogate which represents the behavior of a federation. A Surrogate acquires the information of federation and sends it to the opposite federation. Unlike HLA Bridge, it receives information which cannot acquire from HLA services through direct paths from federates. In addition, Surrogates hold the file which records the information of exchanging objects between federations and use it for data acquisition and filtering.

3.1. Architecture

Figure 3 shows the architecture of 2-way Surrogate. 2-way Surrogate consists of two Surrogates which hold SOMs.

![Figure 3 2-way Surrogate]
Surrogate $S_F$ is a federate representing behavior of federation $F$. Surrogate $S_G$ is a federate representing behavior of federation $G$. Each Surrogate acquires the information of joined federation and sends it to opposite side. $S_F$ joins federation $G$ and acquires the information of federation $G$ from RTI. In addition, $S_F$ is connected to all federates in federation $G$ and acquires their information through these path to solve the HLA problems which is not solved in current RTI. Operation of $S_G$ is same as above except $S_F$ and $F$ are replaced by $S_G$ and $G$.

Each Surrogate holds a SOM. SOM is the file which includes the information of exchanging objects between federations. SOMs of two Surrogates should be same. Surrogates only acquire and exchange the object information recorded in SOM.

2-way Surrogate does not have a transformation manager. It uses object names for mapping entities. Therefore, the same object should be written as the same name in two FOMs.

2-way Surrogate is designed for HLA 1.3 and does not support all services yet. It is designed for data exchange and time management which are required for distributed simulation and for synchronization which is associated with most HLA Bridge problems.

Section 3.3–3.5 explains how 2-way Surrogate works.

### 3.2. Assumption

2-way Surrogate is designed with following assumptions.

1. All federates should send required information to Surrogate of their federation
2. Federations should record the same object with the same name in FOM. Surrogates use this name for mapping an object.
3. Each Surrogate should hold a SOM which includes the information of exchanging objects between federations
4. Partial synchronization cannot be used. Because a federation cannot recognize federations of other federation, selective addressing is not solved.
5. The number of federates should be less than 10 in a federation. If the number of federates is too large, Surrogate should deliver too many information to opposite side.

### 3.3. Data Exchange

In HLA, when data exchange like updating objects or sending interactions is needed, a federate requests an appropriate service with data. Then RTI delivers the information by calling the callback function of received federate. Figure 4 shows procedure of sending an interaction to a federate in opposite federation.

Federate $f_1$ sends an interaction in a federation $F$ using an RTI service. Surrogate $S_G$ receives the interaction via a callback from RTI and delivers it to $S_F$. $S_F$ sends the interaction in a federation. Federate $g_0$ receives it in federation $G$.

### 3.4. Time Management

In HLA, time management consists of two parts. One is delivery of TSO (Time Stamp Order) events and the other is granting a time advance of federates when guaranteeing no events of a smaller time stamp will later arrive [9].

For time managements, HLA uses lookahead and LBTS (Lower Bound on the Time Stamp). Lookahead of a federate is the duration which it cannot send TSO events. LBTS of a federate is the time of the earliest possible TSO events that it can receive. RTI only grants the time advance of a federate less than LBTS for causal behaviors.

A Surrogate uses HLA services for time management. It need not know about the time status of other federates. The Surrogate should equalize its (current time + lookahead) with minimum TSO event time of representing federation. Minimum TSO event time is lower value of LBTS and minimum time of generated TSO events which will be delivered to the Surrogate. `queryMinNextEventTime` service provides that value. Therefore a Surrogate requests this service periodically and if the time value is changed, another Surrogate requests time advance to (Minimum TSO event time – lookahead). Figure 5 shows the procedure of time management. $S_F$ represents the time status of federation $F$.

Surrogate $S_G$ requests `queryMinNextEventTime` periodically. If the time value is changed, $S_G$ sends the value to $S_F$. $S_F$ requests an advance to the time in federation $G$.

---

**Figure 4 Data Exchange**

**Figure 5 Time Management**
3.5. Solution of HLA Bridge Problem

2-way Surrogate uses two methods to solve HLA Bridge problem. One is rigorous implementation and the other is acquisition of required information from new path. Table 1 shows HLA Bridge problems and their solution methods.

Table 1 Solution Methods for HLA Bridge Problems

<table>
<thead>
<tr>
<th>Problem</th>
<th>Federate failure</th>
<th>Service barrier</th>
<th>Consensus</th>
<th>Insufficient information</th>
<th>Selective addressing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solution</td>
<td>Rigorous</td>
<td>Information from new path</td>
<td>not solved</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.5.1. Rigorous Implementation

Some problems can be solved by rigorous implementation. In this case, a Surrogate solves the problems by internal algorithms and uses only the HLA services. Figure 6 shows solution of federate failure.

**Figure 6 Solution of Federate Failure**

Federate $f_1$ requests registration of the synchronization point $L_1$ in the federation $F$. RTIF announces the success of registration to $f_1$. All federates including Surrogate $S_G$ are announced $L_1$. $S_G$ sends the synchronization point $L_1$ to the $S_F$. Then $S_F$ requests registration of the synchronization point $L_1$ in the federation $G$. If RTIG announces the fail of registration, $S_G$ changes the point to $L_2$ and retries registration. If RTIG announces the success of registration, $S_G$ saves the mapping table. All federates are announced $L_2$.

When Surrogates confirms the synchronization point, they use the mapping table. Confirmation of $L_1$ in federation $F$ is same to confirmation of $L_2$ in federation $G$.

3.5.2. Information From New Path

There still remain problems which cannot be solved by rigorous implementation. For these problems, 2-way Surrogate uses two methods. One is holding the required information. This method is used for declaring the exchanging information. The other is acquiring information from federates through direct path.

3.5.2.1 Initialization

In initialization process for interoperation, federates send the initial information to Surrogate through direct paths. For interoperation of federations, Surrogates should join the federations. However, RTI supplies no information until a federate joins the federation. Surrogates will receive the federation information from creating federate. Figure 7 describes this process.

**Figure 7 Initialization**

A federate creates a federation. The federate sends the federation name to Surrogate. Surrogate joins the federation.

When a federate joins and resigns from federation, it also sends the information to Surrogate. If all federates are resigned, Surrogate resigns from the federation.

3.5.2.2 Declaration

In HLA, federates only send and receive object information which is declared previously. To acquire the information for opposite federation, a Surrogate should declare exchanging information between federations. However, RTI does not supply the service for querying the information of other federates.

A Surrogate holds a SOM which includes the exchanging information between federations. When joining a federation, the Surrogate publishes and subscribes all objects recorded in SOM. Therefore, it receives all object information which should be delivered to the other federation.

3.5.2.3 Service Report

When a Surrogate need to know which services federates call, federates send the information to the Surrogate directly. To represent a federation, the Surrogate uses it. Synchronization point achievement and save/restore completion are corresponding services. For synchronization, if all federates achieve the point in a federation, Surrogate of the opposite side federation should achieve the point to represent the federation. However, RTI does not supply the service which announces achievement of a federate to other federates.
Therefore, Surrogate receives the achievement through direct paths. Figure 8 describes the sequence for synchronization. Save/restore process is similar to this sequence.

Each federate achieves a point. The federate also announces the achievement to Surrogate. If all federates achieve the point in a federation F, Surrogate S_G sends the fact to S_F. Then S_F achieves the point.

Using above sequence, consensus problem is solved by following algorithm as shown Figure 9.

Each federate achieves a point in federation F. The federate of the federation F announces the achievement to Surrogate S_G. If all federates achieve the point in the federation F but does not achieve the point in the federation G. This is associated with another problem Service Barrier. To achieve the point in the same time, Surrogate will wait until all federates achieve the point in joined federation. Each federate achieves a point in federation G. The federate of the federation G announces the achievement to Surrogate S_F. If all federates of the federation G achieve the point, S_F sends the achievement to S_G. Then S_F and S_G achieve the point.

4. Implementation of 2-way Surrogate

To verify the interoperation of two federations, we implement the Surrogate. Figure 10 shows the structure of Surrogate. A Surrogate consists of three handlers which handle the information from each path and a manager which stores and manages all information and requests a service to RTI.

Surrogate Manager manages the information from several paths. It requests the appropriate service to RTI by using information from opposite side and database and sends the federation information by using information from RTI and other federates. Surrogate Manager has a database. Required information to represent a federation like object handle map is stored in the database.

Surrogate Handler delivers the information from the opposite federation to Manager and Federate Handler delivers the information from federates to Manager. RTI Callback Handler delivers the information from RTI to Manager or sends the information to opposite side.

Section 4.1–4.3 explains delivery path of information.

4.1. RTI Callback

A Surrogate receives the object information by RTI callback function. In this case, the Surrogate need not hold the information and RTI Callback Handler delivers the information to opposite side. Following sequence represents object update process. Figure 11 shows the data exchange path from federation F to federation G.

RTI Callback Handler of S_G receives the update data of an object. It sends the data to S_F. Then Surrogate Handler of
SF delivers the data to Manager. Surrogate Manager of SF translates the handle of the object and updates the object.

When receiving the callback which announces the registration fail of synchronization point, RTI Callback handler announces the fail to Manager. Then Surrogate Manager will register another point. Figure 12 shows this case. Surrogate SG tries to register a synchronization point Point1 which is registered in federation G.

![Figure 12 Path for registration fail of a synchronization point](image)

RTI announces registration fail of Point1. RTI Callback Handler delivers the information to Manager. Surrogate Manager requests registration of Point2. RTI announces registration success of Point1. RTI Callback Handler delivers the information to Manager and Manager stores a mapping table which translates Point2 to Point1.

4.2. RTI Service

For time management, a Surrogate queries minimum next event time periodically and sends it to the other Surrogate when value is changed. Because the Surrogate acquires the time from return value, Manager sends it to opposite side.

Figure 13 shows data path for time management.

![Figure 13 Path for Time Management](image)

Surrogate SG calls queryMinNextEventTime periodically and compares it with the time stored in database. If the time is changed, Surrogate Manager of SG sends it to opposite side and Surrogate Handler of SF delivers the time to Manager. Depending on time status of SF, Surrogate Manager of SF requests time advance or saves the time.

4.3. Information From Federates

A Surrogate sometimes receives the information through direct path from federates. Services using direct paths need to store information to represent the federation. Therefore Federate Handler always delivers information to Manager. When receiving the information, Surrogate Manager modifies the database, sends the required information to opposite side if needed. Figure 14 is the process for achievement of synchronization point.

![Figure 14 Path for Service Report](image)

A federate of federation F announces achievement of a point. Federate Handler announces the achievement to Manager and Surrogate Manager of SG saves the achievement of the federate. If all federates achieve the point, Surrogate Manager sends the achievement to opposite side. Surrogate Handler of SF announces the achievement to Manager. When all federates of federation G achieve the point, Surrogate Manager of SF sends the achievement to opposite side and announces the achievement to RTI.

5. Experiment

In order to verify interoperability of two federations and measure the performance of Surrogate, we perform the simulations. To verify successful interoperation, the experiment checks data exchange, time management and synchronization. To measure the performance, the experiment measures the message delivery time to other federation.

![Figure 15 System for experiment](image)
For experiment, six user federates are divided into two federations as shown in Figure 15. Each federation consists of three user federates and one Surrogate. DMSO RTI 1.3NGv6 is used for experiment.

To verify successful interoperation, we perform following simulation described in Figure 16.

Figure 16 Experimental simulation

1. Federate A sends an interaction and registers a synchronization point. The interaction contains the random string and the number associated with the point. Federate A records the string to the file.

2. When receiving the interaction, each federate waits logical time 1 and deliver to next federates. The federates also achieves the synchronization point by using the contents of interaction

3. If federate F receives the interaction, update the object by using the contents of interaction and achieves the synchronization point. Federate F records a string of the interaction to the file.

4. When reflecting the object, federate A achieves the synchronization point

5. If federation is synchronized, go to 1. Repeat this process 10 times.

Interactions and the object used in simulation are delivered as TSO event. If time relation is violated, they are not delivered or delivered at incorrect time. Therefore, we compares the time which a federate sends a TSO event with the time which a federate receives the TSO event time. To verify time management, federates records times of all TSO events sent or received. If not synchronized, new delivery process is not started. We identify the synchronization by checking 10 repetitions. If data exchange is performed correctly, sending contents of federate A should be same to received contents of federate F. To verify synchronization and data exchange, we compare the file written by A with the file written by F. Two files are same and record 10 strings.

In order to measure the performance, we measures the message delivery time to other federation. We compares the message delivery time in the same federation with the time across Surrogates.

By varying the number of data at the same time, we measures the delivery time of last data. If waiting time exists due to handling of previous data, the delivery time of last data will be largest. Table 2 and Figure 18 show the results.

Table 2 Delivery time (ms)

<table>
<thead>
<tr>
<th>number of</th>
<th>1</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>50</th>
<th>100</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>path1</td>
<td>31.1</td>
<td>31</td>
<td>31.1</td>
<td>33.1</td>
<td>31.3</td>
<td>31.5</td>
<td>31.2</td>
</tr>
<tr>
<td>path2</td>
<td>117.1</td>
<td>103.1</td>
<td>95.8</td>
<td>90.8</td>
<td>114</td>
<td>96.9</td>
<td>96.4</td>
</tr>
</tbody>
</table>

Figure 18 Delivery time (ms)

When data is delivered through Surrogates, delivery time becomes larger. The reasons are as follows. In the same federation, data is delivered through federate → RTI → federate. However, when data delivered, that process occurs twice and there exists data delivery between Surrogates.

Surrogates also perform the translation of data. If this process takes long time, there exists waiting time. Results show that there is no increment for delivery of larger number of data and we conclude the waiting time is small.

To analyze in detail, we measures delivery time between Surrogates. Table 3 and Figure 19 show the results.
Figure 19 Delivery time between Surrogates (ms)

Delivery time to the other federation should be (delivery time in the same federation) \( \times 2 + (\text{delivery time between Surrogates}) \). Best measurement time is almost same to above expression. However, average value is larger. Because there is a time gap that federates calls tick which is required to receive the data.

6. Conclusion

This paper proposes the architecture 2-way Surrogate for interoperation of federations and implements it. 2-way Surrogate uses a Surrogate which represents a federation and the protocol among the Surrogate and federates to solve HLA Bridge problems. 2-way Surrogate is implemented to verify correctness of the proposed architecture and measure its performance. The experimental results show that the interoperation of two federations is accurate but causes larger data delivery time. Because two RTI divides the jobs, interoperated system will show good performance for the simulation which exchanges a lot of data in a federation. In contrast, it will show bad performance for the simulation which exchanges a lot of data between federations.

Future work would improve performance and design 2-way Surrogate for all services of HLA. Using several Surrogates for a federation consisting of a lot of federates can be one solution of improvement.

7. References


Author Biographies

MIN WOOK YOO is currently a Ph D. candidate at the School of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology (KAIST). He received a B.S in Electrical Engineering in 2007 from KAIST and a M.S. in 2009 from KAIST. His research interests include methodology for modeling and simulation of discrete event systems, interoperation of federations for HLA/RTI.

TAG GON KIM received his Ph.D. in computer engineering with specialization in systems modeling and simulation from University of Arizona, Tucson, AZ, 1988. He was an Assistant Professor at Electrical and Computer Engineering, University of Kansas, Lawrence, Kansas, U.S.A. from 1989 to 1991. He joined at Electrical Engineering Department, KAIST, Daejeon, Korea in Fall, 1991 as has been a Full Professor at EECS Department since Fall, 1998. He was the President of The Korea Society for Simulation (KSS). He was the Editor-In-Chief for Simulation: Transactions for Society for Computer Modeling and Simulation International(SCS). He is a co-author of the text book, Theory of Modeling and Simulation, Academic Press, 2000. He published about
200 papers in M&S theory and practice in international journals and conference proceedings. He is very active in defence modeling and simulation in Korea. He was/is a consultant for defence M&S technology at various Korea government organizations, including Ministry of Defence, Defence Agency for Technology and Quality (DTAQ), Korea Institute for Defence Analysis (KIDA), and Agency for Defence Development (ADD). He is a Fellow of SCS and a Senior Member of IEEE. He developed a tools set, call DEVSimHLA for HLA-compliant war game models development, which has been used for development of three military war game models for Navy, Air Force and Marine in Korea.