2010 3rd International Conference on Computer and Electrical Engineering (ICCEE 2010)
IPCSIT vol. 53 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V53.No.2.50

Distributed Interactive Simulation for Iridium Communication System Based on HLA and OPNET

Yang Yiqun[1+] and Peng Zong [2]

[1]College of Information Science and Technology,Nanjing University of Aeronautics and Astronautics

Nanjing, China

[2]College of Astronautics,Nanjing University of Aeronautics and Astronautics

Nanjing, China

Abstract—Based on the study of HLA (High Level Architecture) and OPNET-HLA interface, the dynamic control of simulation process and the real-time update of simulation for LEO (Low-Earth-Orbit) satellite communication system were achieved on multiple computers. In the distributed interactive simulation, a satellite's location is controlled, then the end-to-end delay time and the routing table of Iridium satellite networks from an earth station to the satellite was calculated with OPNET simulation, and the results were returned to the controller.

Keywords- HLA; OPNET; LEO satellite networks; Distributed Simulation; Interactive;

1. Introduction

In recent years, along with the rapid development of information science and technology, the application domain of computer simulation is expanding intensely , problems to be solved are becoming more and more complex. We often need co-simulation of several computer simulation systems. Therefore, the distributed simulation is widely applied. The High Level Architecture (HLA) software computing developed by US Department of Defense produces many key solutions in management of the distributed simulations, such as interoperability, data distribution management and federation synchronization[1] .

OPNET Modeler is one of the most popular network simulation software. It provides a comprehensive development environment for network modeling and distributed systems. In the previous work, we have built a LEO satellite constellation network by using OPNET Modeler. To control a satellite in real time and calculate the routing table from the earth station to the satellite with OPNET simulation, a method of building distributed interactive network simulation based on HLA is specified in this paper.

The rest of paper is organized as follows: Section 2 gives a brief description of HLA and network simulation software, and presents the OPNET-HLA interface in detail. Section 3 presents implementation of a distributed interactive simulation system for LEO satellite networks. Section 4 is the results of the simulation. Section 5 concludes the paper.

2. Overview of HLA and OPNET-HLA interface

[+] Corresponding author.
E-mail address: yangyq13@163.com

HLA was adopted as an open standard through the IEEE Standard 1516 in September 2000. It provides a common architecture for the reuse and interoperation of simulations. It is a framework to combine simulations into a logical grouping, called a “federation”. Each partner in a federation is called a federate. Federations are typically implemented at a system high classification. Run Time Infrastructure (RTI) is middleware of the software that conforms to HLA Interface Specification by providing necessary services to support HLA compliant simulations. HLA consists of HLA rules, HLA interface specification and object model template (OMT).

There are ten HLA basic rules that define the responsibilities and relationships among the components of an HLA federation. The HLA interface specification provides the requirements of the functional interfaces between federate and RTI. The HLA OMT provides a common presentation for HLA format simulation and federation object models (FOMs)[2]. In this architecture, communication between federates is managed by a MAK RTI. Due to its provision of off-the-shelf tools and toolkits, the MAK RTI is employed in our presented research, which implements the full HLA interface specification, and has been verified by Defense Modeling and Simulation Office (DMSO) as fully compliant with both HLA1.3 and IEEE1516[3].

OPNET is a comprehensive engineering system tool. It is capable of simulating large communication networks with detailed protocol modeling and performance analysis capability. OPNET features include: graphical specifications of models; a dynamic event-scheduled simulation kernel; integrated tools for data analysis; hierarchical and object-based modeling. OPNET analyzes system behavior and performance with a discrete-event simulation engine[4]. OPNET Modeler provides the HLA_interface node, which allows developer to include the Modeler simulation as one of federates in an HLA federation.

By adding HLA capabilities to OPNET Moderler, we can have HLA attribute updates automatically applied to the attributes of a Modeler node, register to receive specific HLA interaction classes and have instances of those classes delivered automatically as Modeler packets, we can also generate HLA interactions, and publish updates to HLA attributes, reflecting changes to Modeler object attributes that occurred as a result of the simulation.

The most important files are *.fed file and class mapping file. The former is shared by all federates. It is based on a comprehensive list of the SOMs of all federates. It lists all object classes, attributes, interactions, and parameters that any federate in the federation can generate, receive, or handle. It does not include type information. A class mapping file specifying translation between[5] the following couples:

- FOM (Federation Object Model) object classes and Modeler object types;
- FOM class attributes and Modeler object attributes;
- FOM interactions and Modeler packet formats;
- FOM parameters and Modeler packet fields;

The format of MAP file used for mapping parameters of OPNET packet to Interaction class parameters is defined as follow:

```
(version
(interactions
(class InteractionName PacketName
(parameter class attribute packet attribute data type)
……
)
……
)
```

3. Implementation of a Distributed interactive simulation system for the LEO satellite networks

In the previous work, we implemented an Iridium network model, which consists of interconnected 66 LEO satellites (plus 6 spares) and the ground-based gateway infrastructure which is required for satellite command and control and connecting them to the ground network system.

Within this work, an earth station subnet(g-2) and a satellite subnet(g-1) are simulated in the Iridium constellation network(see Fig. 1). The orbits of Iridium satellites are created by STK software. In the process of simulation, Iridium satellite nodes fly around the earth on the orbits. The earth station communicates with

g-1 through several Iridium satellites. Since the coverage of Iridium satellite is limited, so a satellite communicates to the earth station will have to rely on the handover scheme and inter-satellite links.

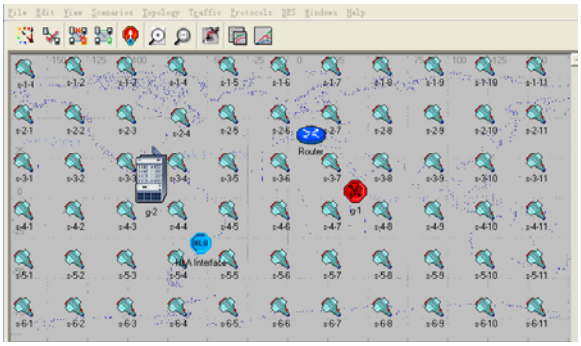

Fig.1. Iridium Network model

The architecture of this distributed interactive simulation is depicted as Fig. 2. The federation name is IRIDIUM_network. It has two federates: controller and OPNET. The controller federate advances federation time, sends interactions to change the orbit parameters of the satellite g-1 in the OPNET federate, then the routing table and end-to-end delay from earth station to the satellite is calculated with OPNET simulation, and the result is returned to the controller.

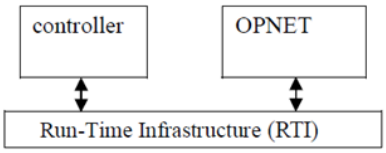

Fig.2. The architecture of a distributed interactive simulation

Two HLA interaction classes are defined in the federation(.fed) file: Order and Order_Reply. The former interaction class includes 3 paraments(latitu, longitu and altitu) and it is mapped in OPNET to Sat_Order packet format, which is composed of 3 fields(latitu, longitu and altitu). Class Order_Reply including 21 paraments is mapped to Sat_Ack packet format(see Fig. 3). They are used to record end-to-end delay time and the orbit plane number and sequence number of Iridium satellites which are passed through. The controller federate pubishes class Order and subscribes class Order_Reply . The OPNET federate pubishes class Order_Reply and subscribes class Order .

The satellite's process model referring to HLA must request interaction classes from the RTI and deliver incoming interaction instances to the appropriate nodes. The process model is modified to receive packets that represent incoming interactions, generate and send an acknowledgment packet. After the satellite process model has received an order to change latitude, longitude or altitude, it should generate an interaction that acknowledges receipt of the order. The interaction returns end-to-end delay time and the Iridium satellites which are passed through when it received the Sat_Order packet.

```
(mapping version 2
  (interactions
     (class Order Sat_Order
       (parameter latitu latitu double)
       (parameter longitu longitu double)
       (parameter altitu altitu double)
     )
     (class Order_Reply Sat_Ack
       (parameter delay delay double)
       (parameter id0 id0 long)
       (parameter id1 id1 long)
       (parameter id2 id2 long)
       (parameter id3 id3 long)
       (parameter id4 id4 long)
       (parameter id5 id5 long)
       (parameter id6 id6 long)
       (parameter id7 id7 long)
       (parameter id8 id8 long)
       (parameter id9 id9 long)
       (parameter id10 id10 long)
       (parameter id11 id11 long)
       (parameter id12 id12 long)
       (parameter id13 id13 long)
       (parameter id14 id14 long)
       (parameter id15 id15 long)
       (parameter id16 id16 long)
       (parameter id17 id17 long)
       (parameter id18 id18 long)
       (parameter id19 id19 long)
     )
  )
)
```

Fig.3. Class mapping file

4. The simulation results

Set up the environment and system variables to access the RTI executive, then run rtiexec.exe and controller.exe, we could see the creation of federation IRIDIUM_Network and the association of the federate controller, finally to run OPNET Modeler in another computer, the OPNET will join in IRIDIUM_Network as a federate. As shown in Fig. 4 below, the two federates: controller(1) and OPNET(2) are running in different computers whose IP addresses are 172.18.15.161 and 172.18.15.183 respectively.

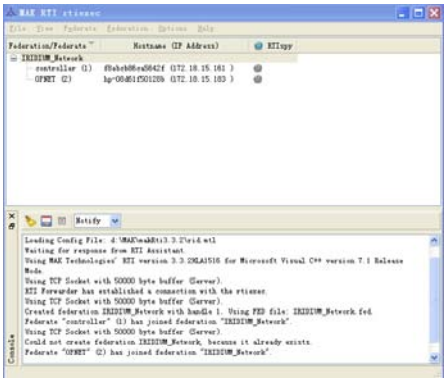

Fig.4. Two federates join the federation

The controller federate advances federation time, sends interactions to modify the longitude, latitude and altitude of the satellite in the OPNET federate, and receives interactions from it. In the controller federate command window shown in Fig. 5, the federation time is advance by entering the 'time' command and the location of the satellite node in OPNET scenario is changed by the 'move' command. First of all, let the OPNET federate advance to time 10s, so there is enough time for all process models to initialize. Then enter 'move' command and the satellite is moved to location A (13 °, 35 °, 510000 m) at time 20s, the three paraments are latitude, longitude and altitude respectively. Continue to advance the federation time to 30s and the result is shown in OPNET simulation Debugger window(see Fig. 6). An interaction is also generated and delivered at time 20s. The interaction is received by the OPNET federate. At time 21s, earth station g-2 at location B (13.3 °, -85.4 °, 0 m) communicates with satellite g-1 via the Iridium satellites s-2-1, s-3-1, s-4-1, and s-5-1. The end-to-end delay is also calculated. At time 21s plus a 'look-ahead time' 0.1s, the OPNET federate should send 'Incoming interaction' back. These data are sent back to the controller federate through HLA and the routing table and delay could be seen in the controller federate command window.

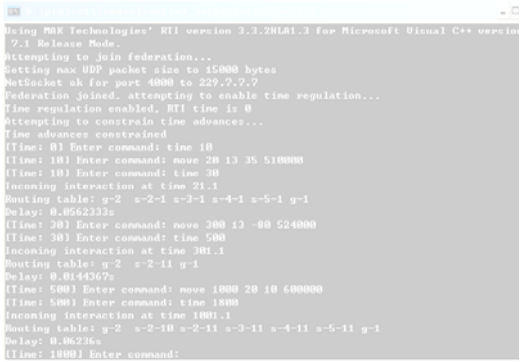

Fig.5. The controller federate sends and receives interaction

Fig.6. The satellite's location is changed at 20s

Then we change the location of the satellite g-1 at 300s and 1000s. The result can also be seen in two federates. It is to be noted that after switching to the location *C* (13^0, -80^0, 524000 m) at 300s, the satellite g-1 will be near earth station g-2 at location *B* (13.3 °, -85.4 °, 0 m). The routing tables between the earth station and the satellite at time 301s and 1001s are shown in Figure 7. It can be seen that the route from g-2 at location *B* (13.3 °, -85.4 °, 0 m) to g-1 at location *C* (13^0, -80^0, 524000 m) only passes through one Iridium satellite s-2-11 in this situation. While there are as many as 5 Iridium satellites to be passed through at time 1001s. The simulation ends at 1800s, as the OPNET simulation duration is set to 30 minutes.

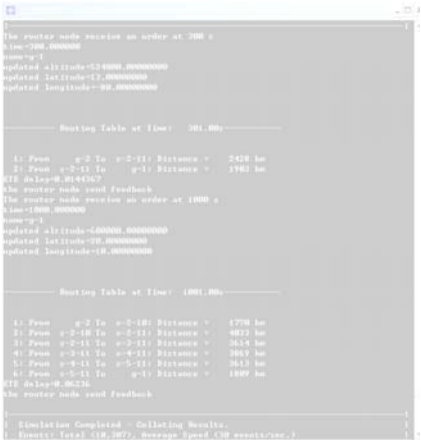

Fig.7. The satellite's location is changed at 300s and 1000s

5. Conclusions

The HLA functionality in OPNET Modeler enables us to include a simulation as one of federates in an HLA federation. In this paper, we introduce the design thought of HLA and the mechanism of OPNET-HLA interface. Then we present the framework of distributed interactive simulation based on HLA for LEO satellite communication system, and construct a simulation federation consisting of two federates, in which the location of a satellite is controlled by another machine and the process is validated by the updated routing table of LEO satellite networks.

Interaction class is taken into consideration in this paper. Actually, object class is often applied to update Modeler object attributes. Mapping the correspondence between HLA objects and Modeler simulation elements, the attributes of a node model in OPNET will synchronously change corresponding to HLA object attributes. The distributed simulation using object class could be implemented similarly.

The simulation results present that the Iridium network model can support the other simulations running on different machines, which means that a large scale of globe network simulation can be done thru HLA over large amounts of distributed machines. With advanced consideration, the simulation might be implemented in more complex and significative scenarios, such as dynamically changing the orbit of a satellite to emulate satellite backup switching.

6. References

[1] IEEE Std1516-2000, "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)-Framework and Rules", 2000.

[2] Frederick Kuhl, Richard Weatherly, Judith Dahmann. An Introduction to the High Level Architecture,[M] National Defense Industry Press, Beijing, June 2003.

[3] http://www.mak.com/products/rti.php.

[4] M. Dooley, J. Dallaire and J. Reaper, "Integrating Behavioral Models with Detailed OPNET Network Models In a Distributed Framework," International Symposium on Collaborative Technologies and Systems, 2003

[5] OPNET Technology Inc. 2004. "OPNET HLA Modeler Use Guide"