Research on Key Technologies of Event-driven Distributed Cooperative Simulation

Wen-guang LI and Song CHENG
School of Aerospace Engineering, Beijing Institute of Technology, Beijing, China

Keywords: Event-driven, formatting, service-oriented, event scheduling, event management.

Abstract. In this paper, an event driven strategy is used to design a distributed simulation system, and a distributed simulation architecture of three-tiers is proposed. Based on the service-oriented design of event-driven distributed simulation system, the hierarchical design of WCF-based simulation hierarchy is completed, and the data transfer and loose coupling between the components of the architecture are realized. Considering the system-level modeling idea based on the design structure matrix, an improved event scheduling method is proposed and the incident management technology is studied. Based on the above research, a simulation system was designed and developed. Taking the multidisciplinary modeling and simulation of the all-electric telecommunication satellite as a case, the simulation application integration and simulation results were analyzed to verify the versatility and correctness of the simulation system.

Introduction

With the continuous improvement of the complexity of modern systems and the continuous diversification of system architecture patterns, the simulation needs and scale are continuously diversified. Therefore, the requirements of simulation for computing resources are also increasing. If the traditional continuous system simulation method is used, the single-time running time is often too long to meet the timeliness requirements of the simulation application, which hinders the development of the application. In contrast, the discrete event simulation focuses only on the events that have an impact on the results and has a high operating efficiency. Therefore, it has become a research hotspot in the field of analysis, evaluation, and argumentation simulation [1].

Discrete event simulation can be divided into time-driven and event-driven methods from the simulation time advancing mechanism. Compared with the time-driven approach, the event-driven approach is more efficient and more applicable to problem-oriented decision-making analysis simulations. Experiments can be conducted efficiently in a relatively large parameter space, and problems can be solved within an acceptable time [2].

The design of modern complex systems often involves multiple disciplines [3]. Due to the complexity of simulation system simulation tasks and the cumbersome process of transaction processing, the running time is very long and the requirements for computer performance are getting higher and higher. To solve this problem, distributed simulation is a feasible solution. Distributed processing technology can manage the computers in the network and integrate their functions to complete the task that a single computer cannot complete, make the decomposition of a large number of simulation tasks into a number of small tasks, shared by multiple computers [4].

Therefore, this paper will discuss the event driven discrete event simulation technology study the key technologies. On this basis, the event-driven distributed simulation system is designed and developed, which has strong theoretical and engineering application value.

Three-tier Distributed Architecture Design

As shown in Figure 1, the composition of the three-tier distributed architecture consists of multiple simulation clients, a master control server and multiple simulation nodes. The simulation client is deployed on the local computer. The server is deployed on the server computer, the simulation node
is deployed on the simulation node machine, and the simulation database is deployed on the database server [5].

![Three-tier distributed architecture topology.](image)

The relationship between multiple simulation clients and the master server and database server constitutes a C/S architecture. The master server is a core component of the C/S architecture and responds to client requests and allocates client access rights. According to request content of the terminal, it operates the database and returns the result of the request to the client.

The main control server and the simulation node constitute a distributed simulation network with a three-tier distributed architecture. Each simulation node machine performs the simulation application and the transfer of simulation results under the control of the main control server. The three-tier distributed architecture enables centralized management of data. The design of direct data transfer between nodes decreases the throughput of the main control server, reduces the operating load of the main control server, and is less complex and easy to implement.

**WCF-based Simulation Hierarchy Design**

In order to make the simulation system meet the requirements of cross-platform and loose coupling, WCF technology is used to build the communication network. According to the system level and logical relationship, the service-oriented idea is used to divide the simulation system into four levels: presentation layer, WCF business logic layer, WCF service layer, and WCF client layer [6]. The system hierarchy diagram is shown in Figure 2.

![Distributed simulation system hierarchy.](image)

Presentation layer: Responsible for the man-machine interaction of simulation process, such as system management, integration of simulation applications, design of simulation tasks, and monitoring and management of node resources.
WCF business logic layer: WCF business logic layer is the core level of distributed simulation system, including simulation development environment, simulation running environment, simulation running auxiliary environment and network communication environment. The simulation development environment is an interactive visualization environment for the design and management of complex simulation tasks. By simulating the interaction between participants and simulation tools, the simulation task design activities are advanced. The simulation running environment is mainly based on simulation applications, managing and controlling the simulation running process and simulation data interaction process. The simulation running assistant environment is mainly responsible for real-time display of simulation process data and evaluation of simulation data.

WCF service layer: This layer provides a service interface for the WCF business logic layer. The WCF client layer interacts with the WCF service layer through the service structure.

WCF client layer: It is mainly responsible for event scheduling and management, driving the operation of models, and data exchange between models.

The WCF server is hosted on the main control server to provide service calls for the simulation node machine. In the WCF duplex mode, the main control server supports the callback of the simulation node machine, so that the main control server can schedule the simulation node machine, send and recycle data.

Research on Event Scheduling and Management Technology

Research on Event Scheduling Strategy

An event is a time-stamped behavior that is triggered by a message and executed by a simulation model. Execution of the event causes the simulation model to change state and dispatch new events [7]. In the simulation system, the implementation of the event is achieved through the change of the output data of the parent model in the two models with data dependency relationships. Therefore, the design idea of the simulation system event model is to consider simulation data and events in a loop to form an improved event scheduling method. The basic flow is shown in Figure 3.

Figure 3. Event scheduling flowchart.
Through the above six steps, a loop of simulation data and event scheduling is formed. In the simulation node, each model corresponds to multiple events (event queues). Therefore, an event belongs to a local member in the system, that is, the event can only affect the model associated with it. At the same time, the model can only respond to its own events. The process of converting the simulation data, event messages and events in the system is shown in Figure 4.

![Figure 4. Event scheduling flowchart.](image)

Before the simulation, the running time of the model is obtained through the DSM solution, and the event is inserted into the simulation node model event queue. In the process of simulation running, the main control server does not need to solve the model running sequence, but only advances the simulation clock. Therefore, this method greatly reduces the scheduling time of the main control server in the simulation running process and increases the operating efficiency.

**Research on Event Management Technology**

Figure 5 shows the flow diagram of the incident management related module. As can be seen from the figure, each model in the simulation node contains an FEL and a PEL. The WCF service receives the event message and submits it to the event generator. The event generator generates the event object through the event message content and the event base class, and distributes the event object to the model associated with the event. The model inserts the event into the FEL to wait for the call. When the simulation clock advances, WCF receives the clock data, scans the FEL of each model through the event scheduler, executes the event with the timestamp as the current clock, and inserts the event into the PEL waiting for recovery after the event is executed.

![Figure 5. Event management process diagram.](image)

In the process of simulation running, frequent scheduling of events is required, including the insertion, extraction, and deletion of events. Therefore, the design of the data structure of FEL greatly influences the operating efficiency of the event, so the data structure of the doubly linked list is used to design FEL and PEL. The schematic diagram of the doubly linked list structure of the event queue is shown in Figure 6.
After the system finishes processing the event, the event is not immediately deleted, but temporarily stored in the PEL for further processing. All processed events will be temporarily stored in the queue. When the GVT advances to a time stamp greater than the time of the event, the system will release the event memory, delete the event, or, if the GVT does not advance to the event timestamp, cancel the execution of the event and then place the event back in the FEL waiting for the next execution.

As shown in Figure 7, when the LVT advances to the timestamp moment of event E, the system will take the event E from the FEL, and then put it into the PEL to wait for processing. After the GVT advances to the timestamp moment of the event, it will delete the event.

System Testing and Application

Based on the above-mentioned research, an event-driven distributed collaborative simulation system was designed and developed. In order to verify the performance of the simulation system in all the aspects studied, now, in view of the demand for the development of complex spacecrafts and based on the simulation and optimization of the all-electric telecommunication satellite, the simulation platform is tested. In this case, the multidisciplinary design optimization problem of AETS satellites can be described as follows: under the condition of meeting design constraints such as time of orbit change, accuracy of position assurance, overall power margin and integral natural frequency, the design variables such as the thickness of composite structural plates and the solar wing area are optimized and optimized to minimize the overall launch quality.

Before the simulation runs, the simulation system inserts events with different time stamps into the model event queue according to the timing relationship of the model operation. Therefore, during the simulation operation, the master server only needs to advance the simulation clock without the need for scheduling. The time-consuming of the event scheduling process will be greatly reduced. The main time-consuming operation is the time-consuming management of the event queue, including the extraction and insertion of events. In order to verify this performance advantage of event scheduling, the following cases are tested.

The number of iterations of the system-level model is 10, 50, 500, 1000, 4000, 10000, and 20000, respectively. The number of iterations corresponding to the model event queue length is 40, 200, 2000, 4000, 16000, 40000, and 80,000, respectively.

After the simulation is completed, statistics of the total event queue length and time-consuming situation of the model during each run are shown in Table 1. The unit event processing overhead is the total time spent extracting an event from the FEL, inserting the completed event into the PEL, and deleting the event from the PEL during the execution of one event, excluding the execution
time of the event. It can be seen from the table that with the length of the event queue increases, the processing cost per unit of event also increases. When the length of the event queue reaches 80000, the maximum unit processing overhead is 215, which has little effect on the simulation running time.

<table>
<thead>
<tr>
<th>Event queue length</th>
<th>Total event processing overhead/µs</th>
<th>Unit event processing overhead/µs</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>6062</td>
<td>152</td>
</tr>
<tr>
<td>200</td>
<td>30748</td>
<td>154</td>
</tr>
<tr>
<td>2000</td>
<td>320642</td>
<td>160</td>
</tr>
<tr>
<td>4000</td>
<td>672084</td>
<td>168</td>
</tr>
<tr>
<td>16000</td>
<td>3034640</td>
<td>190</td>
</tr>
<tr>
<td>40000</td>
<td>8183480</td>
<td>205</td>
</tr>
<tr>
<td>80000</td>
<td>17229120</td>
<td>215</td>
</tr>
</tbody>
</table>

The unit event processing overhead changes with the event queue length as shown in Figure 8. It can be seen from the figure that when the event queue length is less than 4000, the unit event processing overhead increases less. When the event queue length reached 16000, the unit event processing overhead increased significantly, but within an acceptable range. In addition, when the event queue length increases from 1000 to 40,000, the unit event processing overhead increases from 152 to 205, which is a 1/3 increase. Therefore, it can be shown that when the event queue length increases significantly, the increase in unit event processing overhead is less, with good stability.

![Figure 9. Unit event processing overhead changes with the length of the event queue.](image)

Conclusions

In this paper, the function and performance of the simulation system based on the research are tested through a case. The experimental results show that the proposed event scheduling strategy and event management method of the simulation system have good efficiency and stability. On the other hand, the correct operation of the simulation platform verifies the effectiveness of the architecture design and simulation hierarchy design studied in this paper, and provides a reference for the development and application of related distributed collaborative simulation systems.

**References**


