Proceedings of 2012 International Symposium on Information Technologies in Medicine and Education (ITME2012)

August 3-5, 2012

Hokkaido, Japan

Edited by

Xiangwei ZHENG

Xiaohong JIANG

Hong LIU

Peiyu LIU

IEEE PRESS

Home

Proceedings of

2012 International Symposium on

Information Technologies in Medicine and Education

Compliant PDF Files

IEEE Catalog Number: CFP 1253E -ART

ISBN: 978-1-4673-2108-2

Conference CD-ROM Version

IEEE Catalog Number: CFP 1253E-CDR

ISBN: 978-1-4673-2107-5

Print Version

IEEE Catalog Number: CFP 1253E -PRT

ISBN: 978-1-4673-2106-8

Publisher: Institute of Electrical and Electronics Engineers, Inc.

Printed in Beijing, China

A Collaborative Simulation Monitor System of Ship Based on Patterns

Wu Jun
Department of Computer Science and Technology
Jilin University,
Changchun, China
Email: wujun0829@hotmail.com

Liu Shufen
Department of Computer Science and Technology
Henan Polytechnic University,
Jiaozuo, China
Email: liusf@jlu.edu.cn

Li Xiaojun
System Engineering Research Institute
Beijing, China
Email: wujun0829@hotmail.com

***Abstract*—Computer simulation has been widely used in scientific research. But the present simulation systems are mostly designed for a specific purpose, which can only be applied in specific research experiment, so the expansibility is not good. The main purpose of pattern technology is to extend the technologies of software reuse based on Object Oriented [1]. Pattern technology can reuse not only the codes of function modules, but also the structural design of system software. In the design and implementation of the collaborative ship simulation monitor system mentioned in this paper, kinds of pattern technologies have been used. The system has not just the characteristics of high efficiency, easy to extend and maintain, but also the characteristic of high reusability.**

Keywords-Pattern;Collaboration;Simulation;Monitor system

I. INTRODUCTION

With the rapid development of computer science, the computer simulation technology has provided a strong support for scientific research experiment. But the current simulation monitor systems are mostly designed for a specific research task. Thus the application of the system is quite limited. Once a new research task needs to be accomplished, it is necessary to develop a new corresponding simulation system. As a result, a great deal of resources will be expended on developing the corresponding system. Object-Oriented technology of computer science has become more and more mature. The advancement of pattern technology has made us pay more attention to the reusability of the codes when we are developing the system [2, 3]. One of the most important purposes of Object-Oriented technology is to make us better achieve the reuse. Therefore, reasonable patterns mode should be used in the simulation monitor system to make it more flexible in the design and implementation, at the same time to maximize the reusability of the system, and lower the cost of developing in the future. In the design and implementation of the collaborative ship simulation monitor system [4, 5], described in this article, we add in the pattern technology while using the collaboration and simulation technologies, in terms of expansibility and reusability, it has certain advantages. The function modules of the system work together and coordinated to complete simulation monitor tasks through message passing.

II. RELEVANT TECHNOLOGIE

A. *Hierarchical architecture mode*

Component system is based on a specific structure, which is just the architecture. Software system is a single entity that is made up of components. These components constitute the software system with certain architecture, so each of the software systems has a specific architecture. Software architecture defines a system made up of components and their relationships. Architecture is the framework of the software, and it needs to adapt to the changing needs, so we will need a reasonable architecture model to design the software structure. Some of the common architecture models are: broker architecture model, hierarchical architecture model, C/S model and so on. The architecture model used in this article is mainly the hierarchical architecture model [2, 6]. In the system, applied with hierarchical architecture, each layer is a subsystem, and combined together by the layer.

The idea of dividing layer is to break a big problem into some independent subproblems, each layer provides the solution of the corresponding subproblem, reducing the scale

978-1-4673-2108-2/12/$31.00 ©2012 IEEE

and complexity of solving the big problem. The typical hierarchical architecture is: the presentation layer, business layer and data layer. Adjacent layers communicate with each other and work together to provide a total solution to complete the functions of the entire framework. The hierarchical architecture has a good expansibility and the system with it has a high reusability.

B. *Design pattern*

Design pattern is a unified programming method [2], it is a masterly summary of pattern in the design of object-oriented software. It summarizes software design structure, and will be referenced repeatedly in the design of software in the future. It describes the solution to a recurring specific problem. The solution records the component parts of solving a problem, as well as the function of each part and their mutual relationships. There are many kinds of design patterns, each design pattern has its own characteristics and application scope, and it is used to solve the issues with a specific abstract type. The major design patterns are: observer pattern, strategy pattern, composite pattern, command pattern, facade pattern and adapter pattern.

C. *Hierarchical architecture added in design pattern*

Software architecture defines the interrelationships of the components that compose specific software, describes the entire framework, organization, coordination control and other information of the software. Software architecture mode targets a specific domain or problem class. However, design pattern is highly abstract, describes the implementation of the highly reusable components. According to design pattern, we can achieve the reusability of the components and the codes. Design pattern is not for specific problem class, but provides the solutions to abstract problems.

In the hierarchical architecture mode, we add the design patterns based on frequency comparison, such as: command pattern and facade pattern. Due to the frequent operation between the business layer and the data layer in hierarchical mode, when the entity object in the data layer changes, many objects in the upper business layer will be affected. Through introducing command pattern, we can add an entity object of the command pattern between the business layer and the data layer, making the interactions between the data layer and the business layer integrated and forwarded by the entity object of command pattern. Such a mechanism separates the data and business layer, reduces the coupling degree of data layer and business layer.

D. *3D modeling with OpenGL*

OpenGL is actually an interface between graphics and the hardware [7]. It contains a rich and powerful graphics functions, it is not necessary for the developer to write the data of 3D model in a fixed format. Thus the programming is more flexible and the develop efficiency is improved. OpenGL has the following seven functions: modeling, transformation, color mode setting, lighting and material setting, texture mapping, bitmap display and image enhancement, double buffering. The basic flow of OpenGL is as shown in Figure 1.

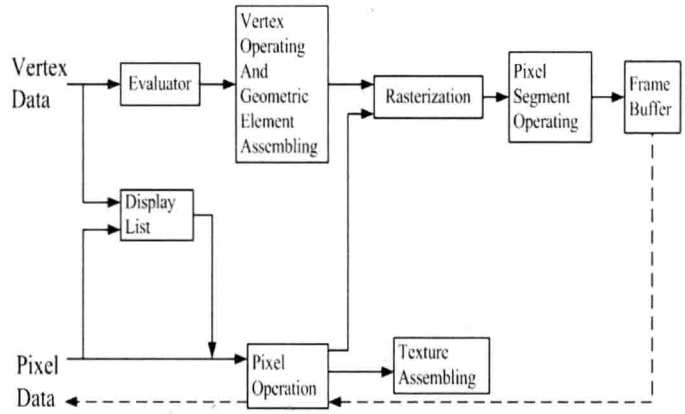

Figure 1. The basic flow of OpenGL

In this paper, we use the OpenGL to complete the 3D simulation of the two observation platforms at both ends of the ship. In order to observe the specified seas, the two observation platforms will adjust their azimuth and pitch angles according to the control message received from the upper layer.

III. SYSTEM DESIGN

A. *Framework design*

The Design of our system architecture is based on hierarchical architecture. During the designing, we bring in the idea of design pattern. The introduction of design pattern makes the system architecture more flexible and efficient, at the same time the reusability of the system is improved.

According to the hierarchical architecture mode, the system is divided into different layers. From the top downward contains the presentation layer, business layer, entity model layer and network communication layer. The system communicates through the interfaces provided by the layers and works together. After introducing the suitable

design patterns, we can improve the communication efficiency and reduce the coupling degree between layers. Thus the framework at all layers can be easily modified and maintained.

B. *System structure and composition*

The main purpose of the collaborative ship simulation monitor system is to get the outside seas information, and visualize the information [8]. So the visualization of the command and control system is enhanced. The framework design has given the general structure of the system. Each layer contains a different function module. These function modules communicate with each other through the network communication coordinated to complete the task. The system is mainly composed of the information transmission module, the surface display module and the 3D simulation model display module. The system structure is as shown in Figure 2.

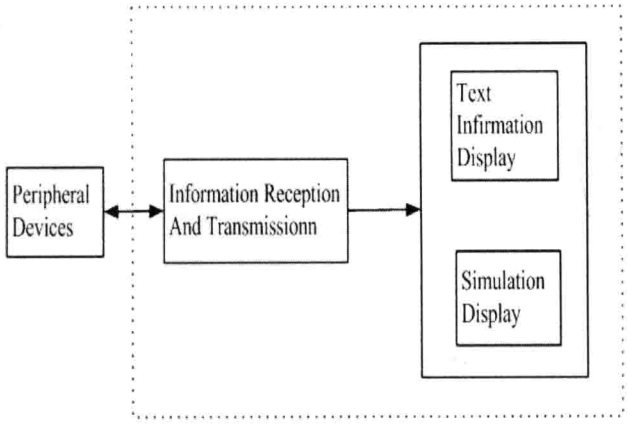

Figure 2. The system structure

IV. SYSTEM IMPLEMENTATION

A. *Implementation of the information transmission module*

The information transmission module is at the network communication layer of the hierarchical architecture mode. The relationship between the communication classes in the layer and the static structure of them is as shown in Figure 3.

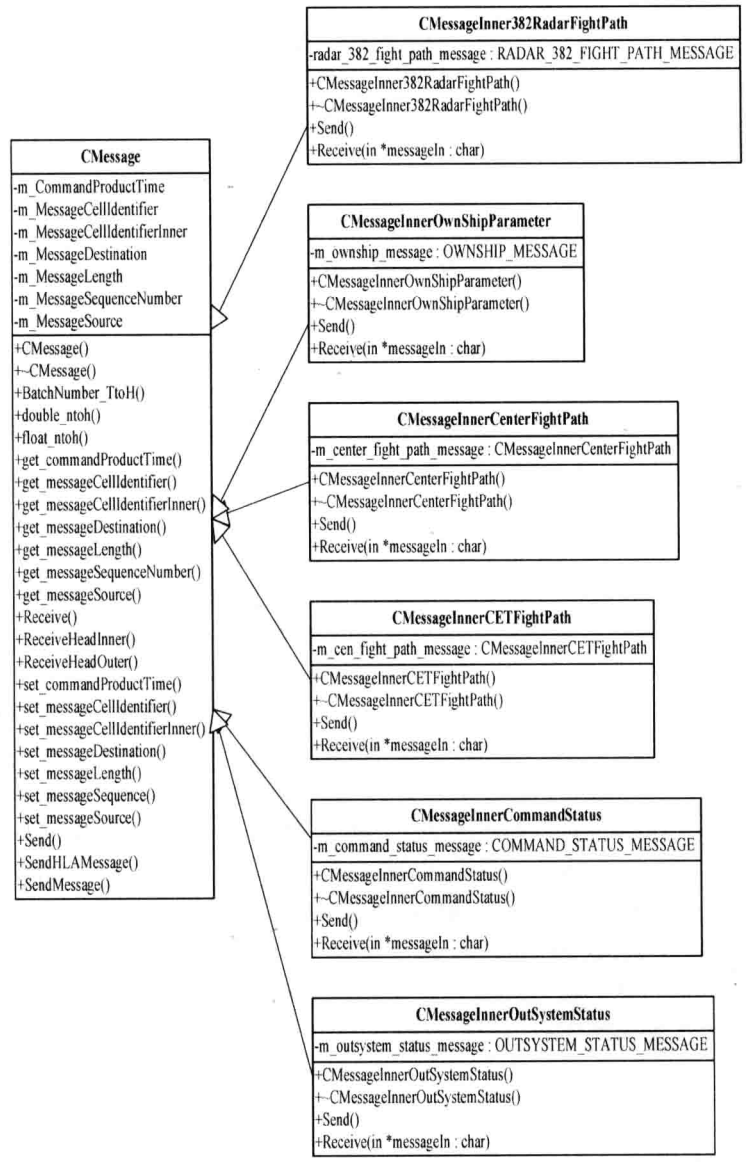

Figure 3. The static structure of the classes in the network communication layer

This module interacts directly with the peripheral devices. The interaction includes both receiving and sending message: On one hand, we can receive the messages sent by the peripheral devices in real time in the simulation test. Through the corresponding network layer class, data in the received messages will be stored into the corresponding entity objects. The function of receiving message can be completed by the function Receive() mentioned in the above figure. On the other hand, in the simulation test, the network communication classes need to send simulated messages to the simulator, and simulate different messages. This function can be completed by the function Send() in the communication subclasses.

The abstract class CMessage needs to create different communication subclasses, depending on different message IDs, and then send or receive messages in appropriate format of the corresponding communication subclass. The design intent of Factory Method is to define an interface used to crate objects, and let the subclass decide which class to be instantiated. In a word, Factory Method puts the instantiation of a class off till its subclass. In the design and implementation of the module, the factory method pattern was introduced. Through designing and building the class InnerMessageFactory (Factory Method) to detect the message identification in the message header, and to specify the object of which communication subclass that we are going to create. Since Factory Method won't bind any classes related to a specific application to the program codes any longer and the codes just deal with the CMessage interface, it can be used together with any communication subclass. The use of Factory Method can also bring the following benefit: Provides a hook for subclasses, it is usually more flexible for creating objects by using Factory Method than creating them directly. Factory Method gives the subclasses a hook to provide an extended version of the object. In the class InnerMessageFactory, we can create the objects of the communication subclass dynamically. Therefore, it is more flexible to send and receive messages and easier to expand and maintain.

B. Implementation of the surface display module

According to the object's property information at the entity model layer and the requirements of the display format, the surface display module needs to show the information to the operator in the form of text. At the same time, it receives the entity information of the simulation model, and displays the corresponding 3D model. The design intent of Observer pattern is to define a one-to-many dependency between objects, once the state of one object changes, all the objects that depends on it will be notified and updated automatically. In view of this characteristic, we use the Observer pattern to complete the module, and it has the following advantages by using the Observer pattern:

1. The coupling between the goal object and observer is abstract. What a goal object knows is just that it has a set of observers, each of which supports the interface provided by the abstract observer class. While the goal has no idea of what a specific class does an observer belongs to. Thus the coupling between the goal and observer is abstract and minimum.

2. Support for broadcast communication. Unlike a usual request, the notification sent by the goal does not need to specify the recipient. It will be broadcast to all the objects that have already been registered to the goal object.

The surface display module is located in the presentation layer of the hierarchical architecture. The presentation layer combines with the Observer pattern to abstract the display class of different types, according to different view display formats. This module is going to create a specific display object through the Observer pattern. In the development of the project, an interface developing plug-in named BCGControlBar is used. According to the operator's operations, the software will select different views. While the building of a specific view object is based on the instructions of the observer object, inheriting a specific subclasses of the BCGControlBar view class. The function of surface display is implemented by calling the interfaces provided by the constructor of the surface located in the business layer. The surface abstract class is built in the business layer, and the business layer will build a specific subclass according to the commands provided by the Command pattern. The business layer first provides the definition of the surface abstract class, and a specific view class will inherit from the subclass of the abstract class. When a command object is passed to the abstract class, the business is going to create the corresponding view class objects depending on the commands. The static structure and the inheritance relationship of the classes are as shown in Figure 4.

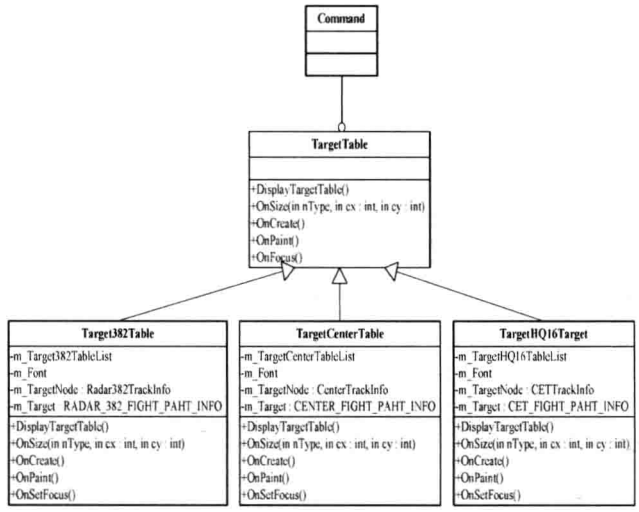

Figure 4. The static structure and the inheritance relationship of the classes of the surface module

C. *Implementation of the 3D model*

The 3D simulation model located in the entity model layer is based on the technology of OpenGL. In OpenGL, the geometry units are all described by vertices, allowing the evaluator and vertex operations to be calculated on each vertex. Then through the rasterization to form graphic pieces[7]. At last, these graphic pieces are sent to the frame buffer after a series of operations to display the graph. GLUT (OpenGL Utility Toolkit) has been used to help draw the 3D model in this module. Since the GLUT is a window system-independent software toolkit and hides the complexity of the APIs provided by different window systems, it makes the application has a high degree of portability. This mode is completed mainly by constructing three classes: class ShipInfo, class ShipEntity and class FormsView. The class ShipInfo stores the parameter information of the 3D model. In the implementation of the module, it receives the related data information from the network communication layer by the interface provided by the class ShipEntity, and then calls the drawing functions implemented in the class FormsView to draw the 3D model. The model in a specific state appears as shown in Figure 5.

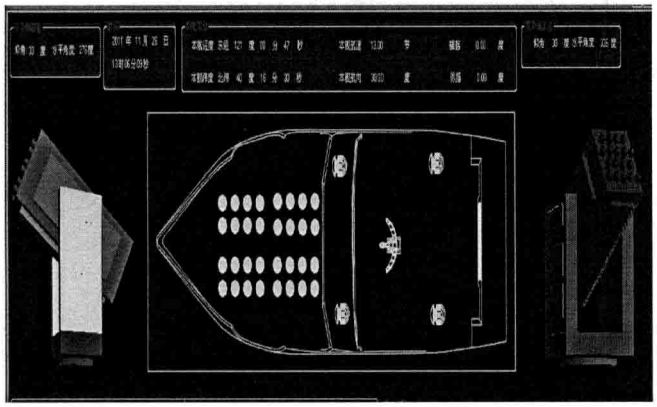

Figure 5. The 3D model in a specific state

In figure 5, the two 3D models located in the left and right sides are the 3D simulation of the two observation platforms at both ends of the ship. The top of the figure displays the real-time information of the observation platforms and the ship. While the middle of the figure displays the ship state according to the received message.

V. SUMMARY

The design thinking of this collaborative simulation monitor system [9, 10] is clear, and the implementation is flexible. Through transmitting information between the various functional modules, they are coordinated to complete the monitoring task, easy to maintain and extend, and with good reusability, which is the main feature of the system. In the design and implementation of the system, a large number of object-oriented technologies have been introduced in, including system architecture mode and design pattern. The hierarchical architecture mode has been used in the design of the framework. An object layer is added between the various layers to support the mutual communications, which reduces the coupling degree of the layers and improves the efficiency of the system. In the implementation of each layer, the technology of code reuse and kinds of design patterns are used. While in the construction of the basic classes, template technology is used, so each function module has a good flexibility and high level of code reuse. OpenGL technology is used in the 3D modeling, which makes the development flexible and fast.

But due to the overall grasp of the design pattern is not yet so good, therefore the introduction of some design pattern is not entirely suitable. In 3D modeling, the texture mapping and lighting of OpenGL still have some room for improvement and optimization. In the future study, I will be more in-depth understanding of design patterns and the related knowledge of OpenGL, hoping to better improve the system.

REFERENCES

[1] Chen Zhaoliang, Wang Qianxiang, Mei Hong et al, Dealing with the Variability in Object-oriented Design [J]. Acta Electronica Sinica, Vol. 29 No. 11, 2001.

[2] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design patterns: Elements of Reusable Object-Oriented Software [M]. Li Yingjun, Ma Xiaoxing et al, translate. China Machine Press, 2011.

[3] S. Srinivasan, Design Patterns in Object-Oriented Framework [J]. Computer, Vol. 32 No. 2, 1999.

[4] Chen Yu, Liu Shufen, Zhu Yongwen, "Framework Design of the Distributed Military Simulation System Based in CSCW" [J]. Ship Electronic Engineering, Vol. 28 No. 1, 2008.

[5] Li Pan, Liu Shufen, Zhang Xinjia, "A Distributed Command and Control Simulation System Framework Based on Design Pattern" [J]. Journal of Jilin University (Science Edition), Vol. 46 No. 3, May 2008.

[6] Wang Qiyuan, "The latest Method of Simulative Monitor Control System the Research of Three-level Client/Server Model" [D]. Wuhan: Wuhan University, 2006.

[7] D. Shreiner, OpenGL Programming Guide [M], 7th ed. Li Jun, Xu Bo et al, translate. China Machine Press, 2010.

[8] Sun Weiqiang, Pi Yiming, Cao Zongjie, "A Monitor System for Airborne Fire-Control Radar Simulation System Based on HLA" [J]. Vol. 26 No. 11, Nov. 2009.

[9] Cao Wei, "Computer Supported Cooperative Design System: Integration Model and Realization Method" [J]. Journal of Eastern Liaoning University (Natural Science), Vol. 17 No. 1, Mar. 2010.

[10] Zheng Hongbo, "The Research and Implementation of Computer Supported Cooperative Design System" [D]. Changchun: Jilin University, 2004.