SERVICE ORIENTED MINE HUNTING CLASSROOM SIMULATION SYSTEM

MIKUS VANAGS, AGRIS NIKITENKO, MARTINS EKMANIS, ILZE ANDERSONE, ILZE BIRZNIECE, GUNTIS KULIKOVSKIS

Riga Technical University, Department of Systems theory and design, Meza str. 1/4, Riga, Latvia, LV1048, email: vanags.mikus@gmail.com

Abstract

It is well known that mine hunting ship crew training on a real ship is expensive. More cost – effective way is to use a software-based classroom simulator. The purpose of this paper is to describe architecture of Alkmaar class mine hunting ship classroom simulation system built for Latvian ministry of defense. Simulation system is built as a service oriented system and enables high connectivity with other systems – simulators. Computer graphics based user interface ensures high systems flexibility and high quality human-system interactions. Use of object database eliminates the need to predefine or maintain a separate, rigid data model leading to reduction of system development time and database maintenance costs. The simulator is driven by three mathematical models: weather, ship and robot mathematical model. Our work contains analysis of the best practices of designing and implementing service oriented classroom simulator – it's mathematical models, service oriented architecture, database design, ways of user interactions perception, user interface visualization and physical implementation using bleeding edge technologies like .NET 3.5, Windows Communication Foundation, XNA Game Studio and other.

Introduction

It is well known that mine hunting ship crew training on a real ship is expensive. More cost – effective way is to use a software-based classroom simulator. The simulation system is designed for Latvian Ministry of defence to emulate Alkmaar class ship and mine hunting robot Pap104 controlling process during mine hunting operation in operations center (OC) as well as to log all user interactions and analyze accomplishments of the crew.
The simulation system consists of sonar control panel, two sonars (detection and classification sonar), plotting table - computer with integrated geographic information system, ship control panel, mine hunting robot control panel and measurement displays. Team with up to four crew members can participate in mine hunting operation simulations.
The paper sections are organized as follows. First the mathematical models are described. Then analysis of event mechanism design in service oriented systems and demonstration. Third section deals with implementation of sonar ray tracing and visualization is presented and forth is devoted to physical model description of simulation system. Results, discussions and conclusions follow afterwards.

Mathematical models

The simulator is driven by three mathematical models: ship's model that allows to simulate 6 DOF correlated with weather mathematical model i.e. including waving, water laminar resistance and side winds; robot's model in its essence is simplified ship's model that does not need some of the weather components; weather model allows to simulate different weather conditions starting from constant for simpler scenarios and ending with constantly changing with random events such as sudden wind blows for complex scenarios.

Service oriented architecture

The key feature of this classroom simulation system is its service oriented architecture, where the most important feature is event handling and routing in distributed environment. We tried two different technologies: .NET Remoting (Kimmel, 2005) and WCF 3.5 (Chakkaradeepcc, 2008).
Figure 1 shows one way how to implement event handling mechanism using .NET Remoting tehnology. First, the listener subscribes to particular event through remote object proxy. Second, initiator initiates the event and third, event handler on listener side gets the event notification. The problem is that Handler declaration needs to be publicly shared across all simulation system components. Such restriction makes the simulation system very difficult to maintain and adapt to any changes that might occur.

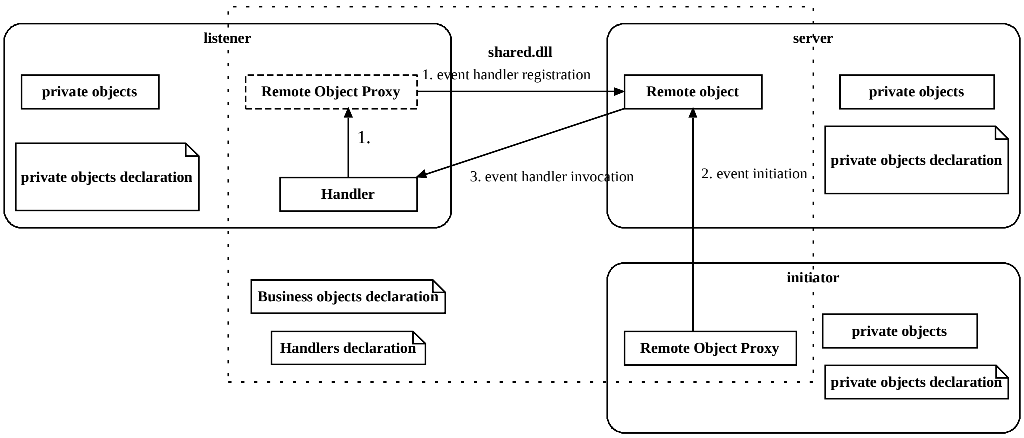

Figure 1. Simple .NET Remoting event handling scheme with public event handling methods

The solution to exclude private event handlers from publicly shared dll (Dynamically linkable library) is the use of event wrapper principle as depicted in figure 2.

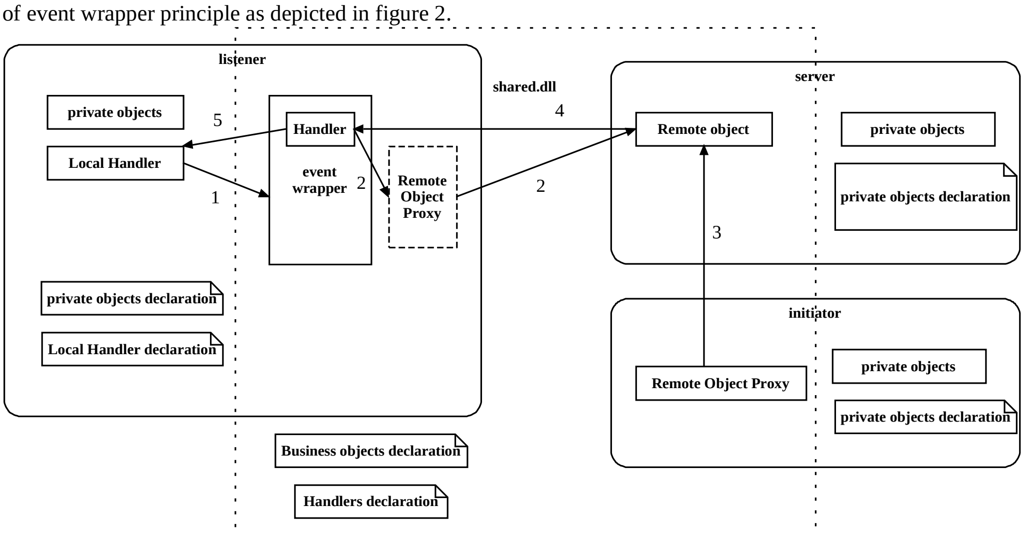

Figure 2. .NET Remoting event handling scheme with private event handling methods

There are two event layers. The first one is event wrapper layer, where local event handling methods are registered and the second one is subscription layer to 'Remote object' event, where event wrapper method named 'Handler' is registered. In this way event the wrapper objects are created on client side and server side gets reference to client object accordingly. When event occurs on the server, client will execute appropriate event wrapper 'Handler', which will call appropriate local event handler.
The event handling mechanism can be described in 5 steps:

1. Creation of the event wrapper object and local method named 'Local Handler' registration to receive event wrapper events.
2. Registration of event wrapper method named 'Handler' to respond to Remote Object events and raise event wrapper events.
3. Initiation raises Remote Object event.
4. Execution of Remote Object event handler – event wrapper method named 'Handler'.
5. Method named 'Handler' raises event wrapper event to which local event handling methods were registered and these methods are now invoked.

Event wrapper solution had unacceptable performance and as the alternative was decision to implement service architecture using WCF technology which is shown in figure 3 (Chakkaradeepcc, 2008).

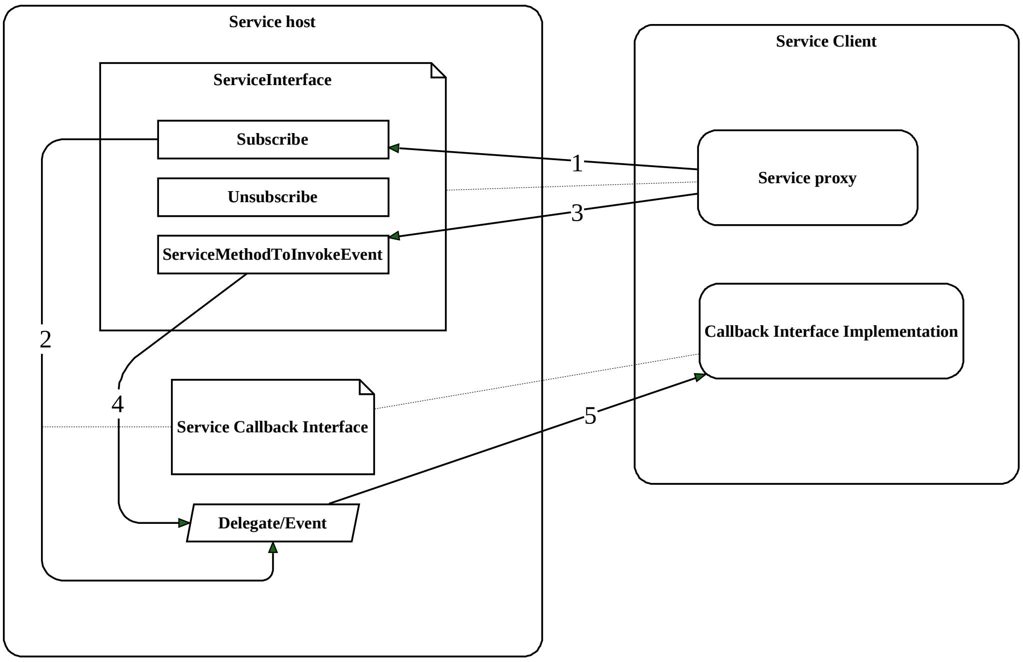

Figure 3. Event handling mechanism using technology WCF

Steps of event handling mechanism depicted in figure 3 are the following.

1. The client calls service method ‘Subscribe’.
2. Subscribe method associates callback interface methods that exists on client with appropriate service events.
3. Client calls service method which raises the event.
4. Event underlying delegate is invoked.
5. Delegate invocation executes appropriate callback interface methods on client.

Figure 4 shows the simulation system application layers designed for WCF event handling mechanism. Business logic is implemented in the business layer, while business layer interaction interface is exposed to other simulation system applications through Windows Communication Foundation services.

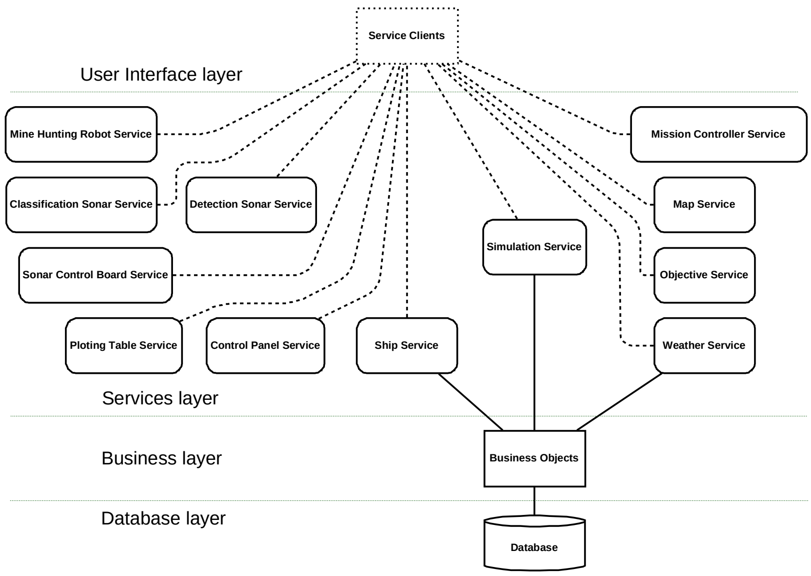

Figure 4. Simulation system application layers

While database, business and services layers are located on simulation server, user interface layer consists of several parts, each of them implemented as a different application. Thereby changes in the business logic don't impact user interface layer simplifying future enhancements and maintenance. Besides, in case of failure in any user interface application the rest of the simulation system applications continue to run. A restart of any user interface application can be done without stopping simulation process or corrupting data, because simulation data and business processes are hosted on the simulation server.

Sonar ray tracing and visualization

The central visible part of the simulation system is detection and classification sonars that are implemented using Microsoft XNA Game Studio 3.0 (this tool is useful in computer game creation).
High Level Shader Language (HLSL) is used as a primary tool for massive parallel computing on graphical processor. For real-time sonar ray tracing it is important to make huge amount of simple and independent calculations in parallel. Sonar visibility range can reach 1km and more, hence at the resolution set to 50cm we should actively render $1.6*10^7$ terrain elements each 16ms (60 updates per second). Such amount of calculations is impossible to make even on the latest GPUs.
To solve this problem we have implemented Clipmaps (Asirvatham and Hoppe, 2004). Our clipmap implementations have symmetric layers allowing recursive rendering. Each next level have higher details and changed scale, allowing higher level of details closer to source of sonar signal, and lover details in the distance (see figure 5).

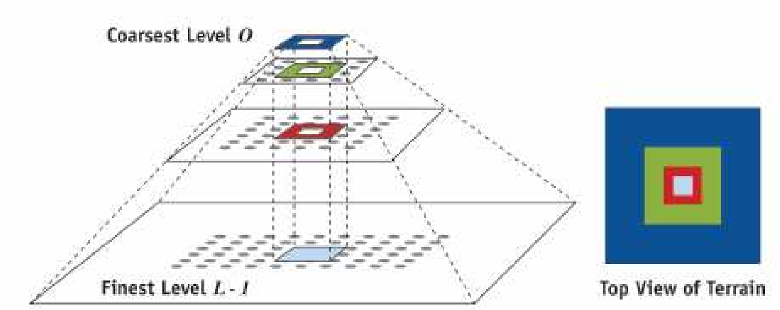

Figure 5. Principle of clipmaping (Asirvatham and Hoppe, 2004)

In clipmap the most important is symmetry in structure and border areas. Examples of clipmaps with different layers are depicted in figure 6 and figure 7.

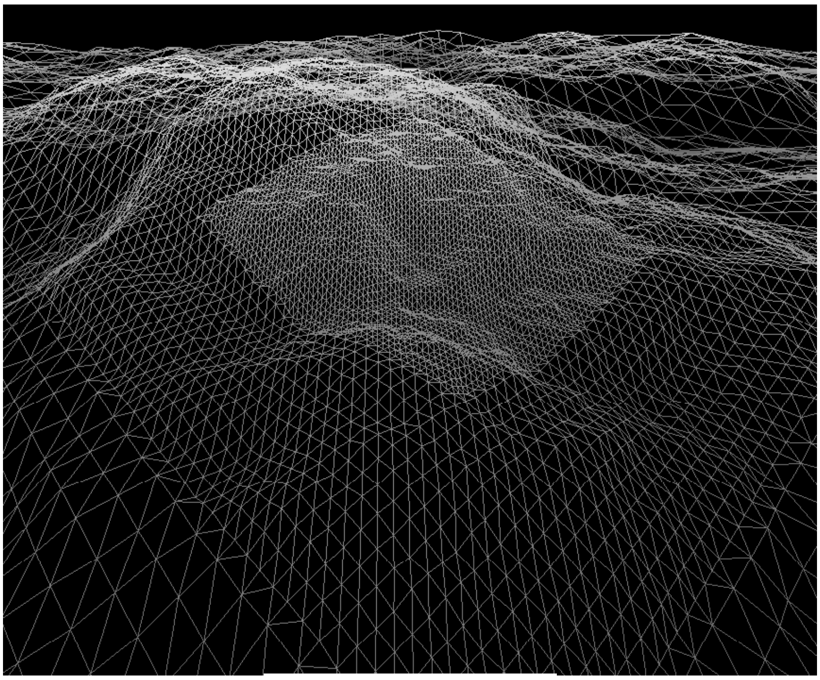

Figure 6. Clipmap 57 x 9 Layers

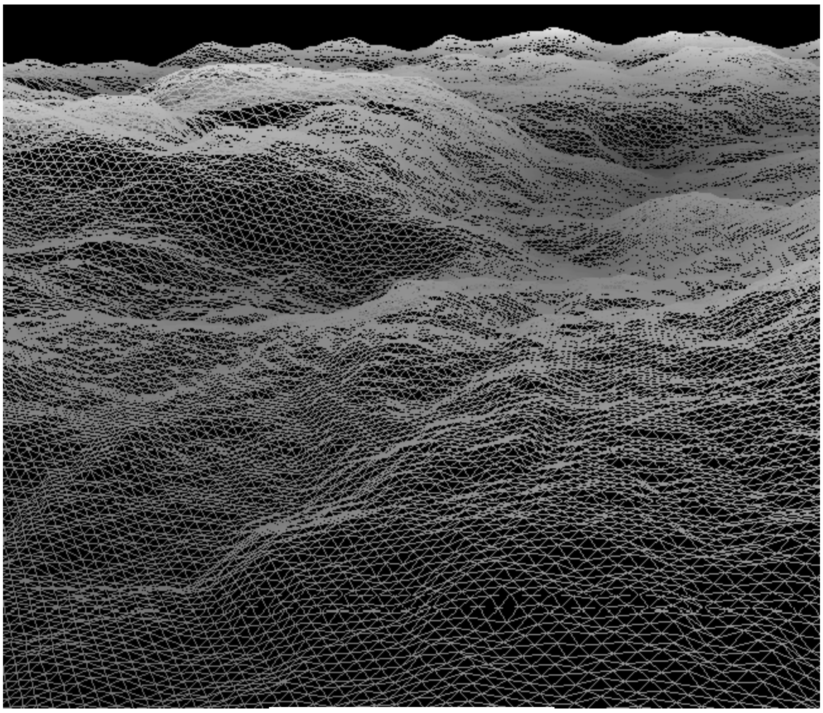

Figure 7. Clipmap 253x5 layers

Clipmap operations like world coordinate, signal source and ship transformations are performed on vertex shaders. Ray tracing and shadow calculations in the same time are calculated on pixel shaders. Actual complexity of our solution requires only ~ 600 million floating point operations per second, what is reasonable load for average office PC.

Physical model

Classroom simulation system physical model is built as a simple LAN (see figure 8). Sonar displays and other devices are reproduced as standard displays or touch screens. Therefore in case of damage every simulation system component can be easily replaced.

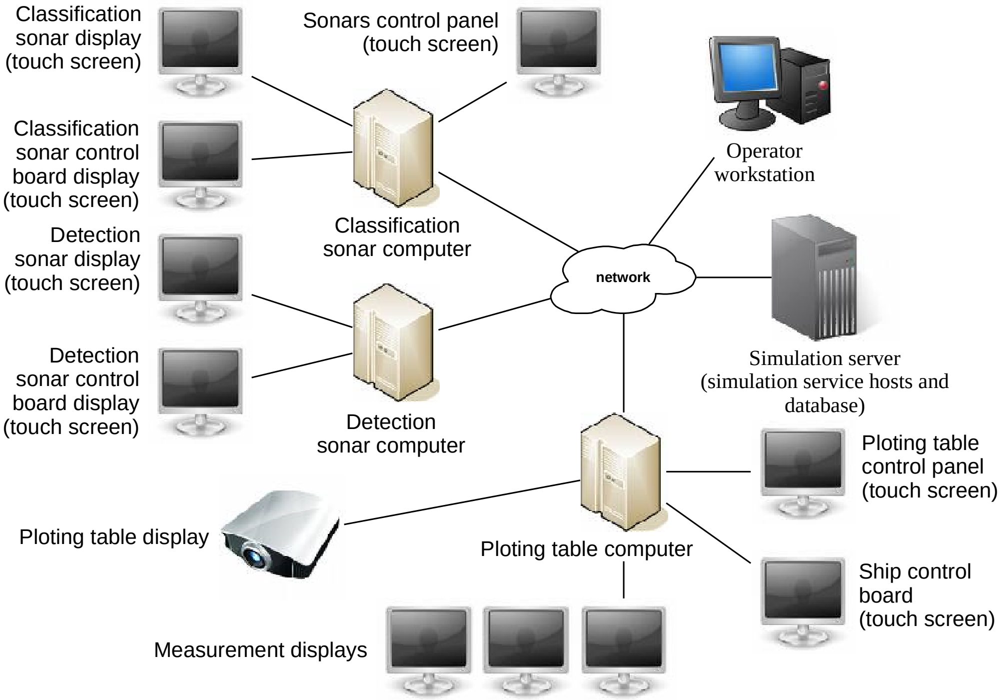

Figure 8. Simulation system physical model

All simulation system displays are designed as touch screens transferring responsibility of user interface design and implementation to software development team. Thereby necessity for other specific user interface engineers is eliminated and system maintenance costs are reduced. Other user interaction objects are industrial joysticks and switches intended for ship and mine hunting robot control.

Maps, missions and all simulation logging data are stored in the open source object database Db4o that enables .NET developers to store and retrieve any application object. Native object database eliminates the need to predefine or maintain a separate, rigid data model leading to reduction of system development time and database maintenance costs. Object database also improves simulation system performance by reducing type conversion operations that would occur in alternative solutions like in case of object relational mapping.

Results

We have developed three mathematical models to drive simulation: weather, ship and robot mathematical model. Simulation system architecture is service oriented and software components is developed using .NET Framework 3.5 and simulation system physical model is built as a usual computer network. Sonar displays and other devices are reproduced as standard displays or touch screens.

Discussion

Classroom simulation system is built as service oriented system and enables high connectivity with other systems – simulators. Computer graphics based user interface ensures high systems flexibility and high quality human-system interactions. Use of object database eliminates the need to predefine or maintain a separate, rigid data model leading to reduction of system development time and database maintenance costs.

Simulation system is implemented using cutting edge technologies like .NET 3.5, Windows Communication Foundation, XNA Game Studio and other. Physically simulation system is developed as usual computer network, therefore development, maintenance and potential enhancements of simulator are cost effective.

Conclusions

The selected technological platform in combination with the developed architecture provides a reliable solution for massive real time calculations as well as high modularity and maintainability.

Service oriented systems with event handling mechanism are not reasonably to implement using technology .NET Remoting because of low event handling performance. Much easier and better from performance perspective is usage of Windows Communication Foundation.

Our clipmap algorithm implementations have symmetric layers allowing recursive rendering which leads to reasonable load for average office PC for ray tracing and sonar visualization.

Acknowledgements

The presented results are part of the project with reg. num. AĪVA 2007/227 funded by Ministry of Defense of Republic Of Latvia and fully implemented by Riga Technical University

References

1. Asirvatham A., Hoppe H. (2004) Terrain Rendering Using GPU-Based Geometry Clipmaps. Available at: http://research.microsoft.com/en-us/um/people/hoppe/gpugcm.pdf, 15.02.2010.
2. Chakkaradeepcc. (2008) WCF Implementation of the Publisher/Subscriber Model. Available at: http://www.codeproject.com/KB/WCF/publisher_subscriber.aspx, 15.02.2010.
3. Kimmel P. (2005) .NET Remoting: Handling Remote Server Events. Available at: http://www.informit.com/articles/article.aspx?p=366699, 15.02.2010.