MAÇA YAZILIM

ONLINE VIRTUAL TEAM COLLABORATION PLATFORM WITH 3D GRAPHICS

CENG 492

SpadeShip Developer's Guide

METU

2008

Table of Contents

1. INTRODUCTION 3

1.1 SpadeShip Project 3

1.2 Simulation Scenario 3

1.3 System Information 3

2. SYSTEM AND TOOL CHOICES 3

3. ARCHITECTURE 4

4. DIRECTORY STRUCTURE 5

5. COMPILATION INSTRUCTION 5

6. CHANGING GRAPHICAL USER INTERFACE 5

7. CHANGING CHARACTERS & ADDING NEW MODELS 5

8. NEW MAPS 6

1. INTRODUCTION

This document is prepared for developers who want to change or develop Spadeship project. Further information about classes and methods (API) can be found as doxygen documentation.

1.1 SpadeShip Project

Spadeship is an educational virtual team collaboration platform with 3D graphics that simulates an emergency situation.

1.2 Simulation Scenario

Spadeship simulates a fire situation in a passenger ship. There are four users (roles) to handle this situation and save passengers and the ship with collaboration. In short the user with fireman role uses his/her resource to extinguish the fire and allow others to pass safely. The medical chief uses his/her resources to heal the injured passengers, and after that captain direct or lead the healed passengers to the safe areas in the ship. Lastly facilitator watch the simulation, interfere by creating more fire and injured passengers and send resources to fire chief and the medical chief when asked.

1.3 System Information

The system is based on client/server model. The facilitator is actually the server and the other users are the clients. All communication (included voice communication) is broadcasted by the server to the clients. Both the server part and the client part are developed under Windows.

2. SYSTEM AND TOOL CHOICES

System and tool choices can be grouped into four broad categories:

Operating System Choice: Windows XP Operating System.

Hardware Choices: P4 class processor or equivalent, 256MB of memory, Graphics card and Direct3D support, sound card, internet or network connection, devices for voice

communication.

Open Source Engine Choices:

• Graphics Rendering Engine: OGRE, one of the most popular rendering engines, is

chosen for the project. Most important features of OGRE that affected the choice were its design quality, flexibility and clear documentation. As the sample programs using OGRE were really helpful in developing the project, using OGRE brought some additional advantages. The choice of OGRE affected the decisions of input handler and GUI library directly. The OIS (Object Oriented Input System) and CEGUI respectively for the input handler and GUI library are chosen for their compatibility with OGRE.

• Network Engine: OpenTNL (Torque Network Library) is chosen as the network engine. It was allowing the developer to transfer the objects as argument, string,

byte buffer, bit stream or vector. It has a supportive documentation and samples.

• Sound Engine: OpenAL and DirectSound will be used for sound. OpenAL was preferred for playing the sounds on the other hand DirectSound was preferred for recording the voices.

• Voice Codec: LPC10 of Hawk Voice was chosen to encode and decode the recorded sounds for its high performance on compression of the voice.

• Physics Engine: ODE is widely used with OGRE, so we have decided to use it as the physics engine.

3. ARCHITECTURE

The server initiates the simulation and waits the clients to connect. After all clients are connected the server starts the simulation. All physics calculations and AI decisions are done by clients separately rather than broadcasting by the server to decrease latency, but from time to time this information is synchronized. On the other hand voice messages are first sent to the server and then server broadcasts them to the clients. At

the end server (facilitator) stops the simulation and all the users pass to the evaluation screen. Every user included facilitator rates all roles, server collects this information, calculates the scores and sends back this information to all roles. Then all users terminate the simulation separately.

4. DIRECTORY STRUCTURE

In Spadeship directory there are doc, media and release directories and source code files. In doc directory there are documentation files: user's manual, installation manual, developer's manual. In media directory there are model files, texture files and map related files. In release directory there are used libraries and the executable file.

5. COMPILATION INSTRUCTION

Microsoft Visual Studio is used as development tool so Spadeship.sln can be used to open the project from Visual Studio and after exporting project settings from using Spadeship.vsproj file and adding libraries that are in Spadeship Dependencies, the project can be compiled and run with VisualStudio.

6. CHANGING GRAPHICAL USER INTERFACE

To make changes or adding new menu's to Spadeship project, Spadeship.layout file must be edited by hand or by using CEGUI editor. This file can be found in Spadeship\gui\Media directory. If menu image will be used these image files must be put into Spadeship\Media\gui\image directory.

7. CHANGING CHARACTERS & ADDING NEW MODELS

To change character models, the model files in Spadeship\Media\models directory must be changed. All the characters are in OGRE .mesh file format (captain.mesh, medic.mesh, fireman.mesh). This charater's .skeleton files are also in the same directory and the .material files should be put in Spadeship\Media\materials\scripts directory. Texture files should be put in Spadeship\Media\materials\textures directory. There are

various OGRE XML exporters for many modeling tools. After exporting .xml files OGREXMLConverter can be used to convert them to .mesh files which can be obtained from OGRE website as OGRECommandLineTools. Also new models can be added by following the steps stated above.

8. NEW MAPS

The map used by Spadeship is in .bsp file format. But the map has been developed as Quake 3 map with GTKRadiant editor and after that it compiled to .bsp by using Quake Tools q3map. Then the .bsp file and used textures must be zipped and put in Spadeship \Media\packs directory.