Multimedia Systems –
C-WG3#1
Design Specification

PURPOSE OF THIS DOCUMENT

This document contains everything concerning design and implementation of the course work.

REVISION HISTORY

Revision #	Date	Author(s)	Comments
0.1	09.11.09	Pasi Keski-Korsu	Template created
0.2	09.11.09	Pasi Keski-Korsu	Notes added
0.3	30.11.09	Pasi Keski-Korsu	Text and pictures added to chapter 5
0.4	30.11.09	Pasi Keski-Korsu	Chapter 4 written
0.5	30.11.09	Antti Väyrynen	1&3 added
0.6	30.11.09	Pasi Keski-Korsu	Table of contens and small fixes
1.0	30.11.09	Lauri Majamaa	Final version, converted to docx & pdf

DEFINITIONS AND ACRONYMS

UDP	User Datagram Protocol
RTP	Real-time Transport Protocol
GLC	OpenGL & ALSA video capture tool for Linux
FFmpeg	Cross-platform solution to record, convert and stream audio and video
ACK	Acknowledgment
Qt	Cross-platform application development framework
LibVLC	Video decoding and playing framework
Git	Fast Version Control System
OpenGL	Open Graphics Library

TABLE OF CONTENTS

1 INTRODUCTION

The project is to create a working example of cloud gaming. In principle, cloud gaming means that the game runs on a server and the player interacts only with the client machine. We decided to have Linux on the server and Windows on the client machine. The server will be running the game, video streaming scripts and software to receive game controls. Game video is captured and then streamed to client. User controls are then send back to the server by the client program. In order to have an enjoyable game experience the time difference between video and controls has to be minimal.

For capturing video from the game itself we chose to use an open source capture framework GLC. GLC capture works only with games that use OpenGL graphics to draw frames and is normally used only to record video, not for streaming it. Raw video from GLC's output is then compressed using FFmpeg and streamed with its FFserver. Stream uses real-time stream transport protocol RTP, which functions on top of UDP. RTP has several advantages over alternatives because of its real-time nature, and lower buffer can be used. Video is compressed as H263, a light weight codec designated to video conferences. Both GLC and FFmpeg were chosen because of their simplicity and sufficient documentation that allowed modifying them to our distinct purposes. We chose not to use Yukon framework for capture, like was originally planned because of its bad streaming performance.

Client software is implemented on top of Qt, a cross-platform application development framework. Qt provides easy-to-use graphical user interface (GUI) and is able to show video with LibVLC libraries and send game controls to server. In server end, software that receives the controls is also written on top of Qt. The cross-platform framework allows us to port the client both to Windows and Linux without problems. LibVLC is a framework for Videolan Client, a popular cross-platform media player, but it can also be used standalone with our client software. It provides tools for receiving and decoding the video stream, and as well displaying it in the user interface.

In this project we are using only UDP network protocol because of its time-sensitive nature. The protocol is simple and has no handshaking or means to determine packet loss between client and server. Therefore it should provide faster performance than traditional streaming on top of HTTP which uses TCP. Real time controls are also a good excuse to use UDP. For the version management we chose to use Git.

2 FRAMEWORK AND PROTOCOLS

Qt is a cross-platform application development framework.

GLC is an ALSA and OpenGL capture tool for linux.

RTP defines a standardized packet format delivering audio and video over the Internet. RTP is used extensively in communication and entertainment systems that involve streaming media.

UDP is network protocol used to send datagrams to other hosts on an Internet Protocol network without requiring prior communications to set up special transmission channels or data paths. UDP uses a simple transmission model without implicit hand-shaking dialogues for guaranteeing reliability, ordering or data integrity.

OpenGL is a standard specification defining a cross-language, cross-platform API for writing applications that produce 2D and 3D computer graphics.

LibVLC is a library for VLC media player. It can be embedded to an application to gain audio/video playing features.

XTest is an extension library for X.org X11 server. It is a minimal set of client and server extensions required to completely test the X11 server with no user intervention.

3 DESIGN LIMITATIONS

Our program runs on two different computers. Client computer does not have huge limitations on hardware or software. Well, that's the idea of cloud gaming! Nevertheless, server computer needs quite a lot processor capacity. Running the game itself needs a lot resources and added to that server must, among other things, compress the video. One particular task is not heavy to process but when there are lots of small tasks the total need of resources is quite high.

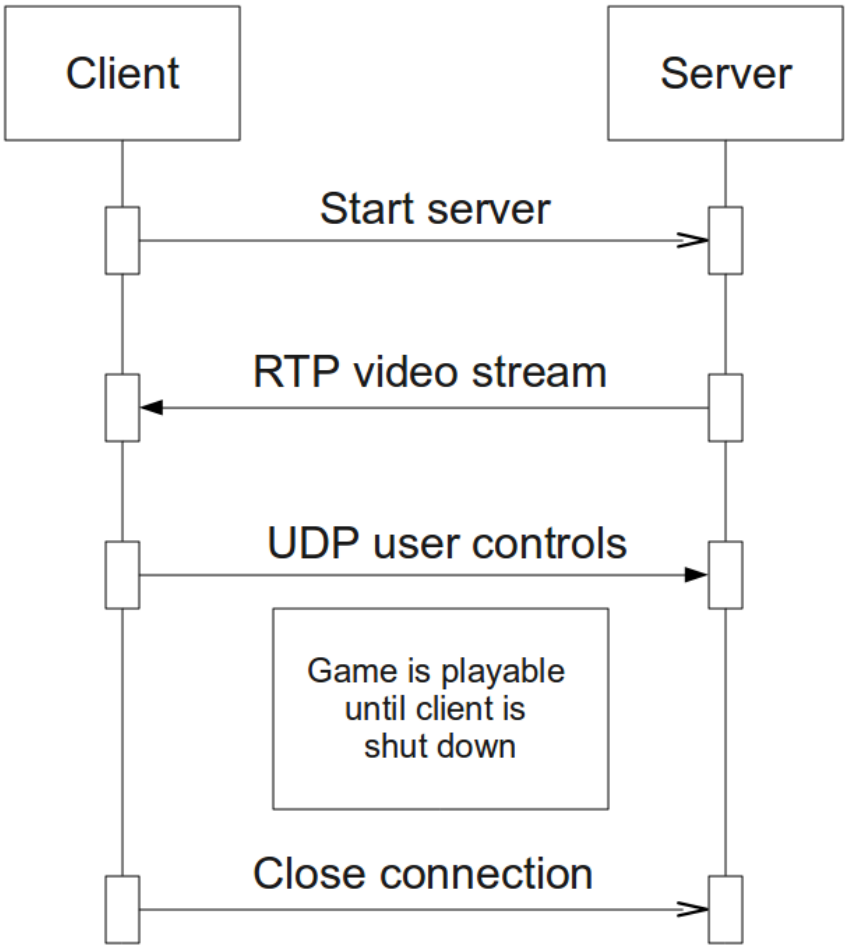
Second main limitation in our project is network delay. Any kind of network is usable to our program but on testing we use Ethernet cable. Ethernet cable is the most efficient way to connection between server and client. WLAN or Internet may cause network delay which is not depended on our system.

4 PROJECT DESIGN

4.1 General

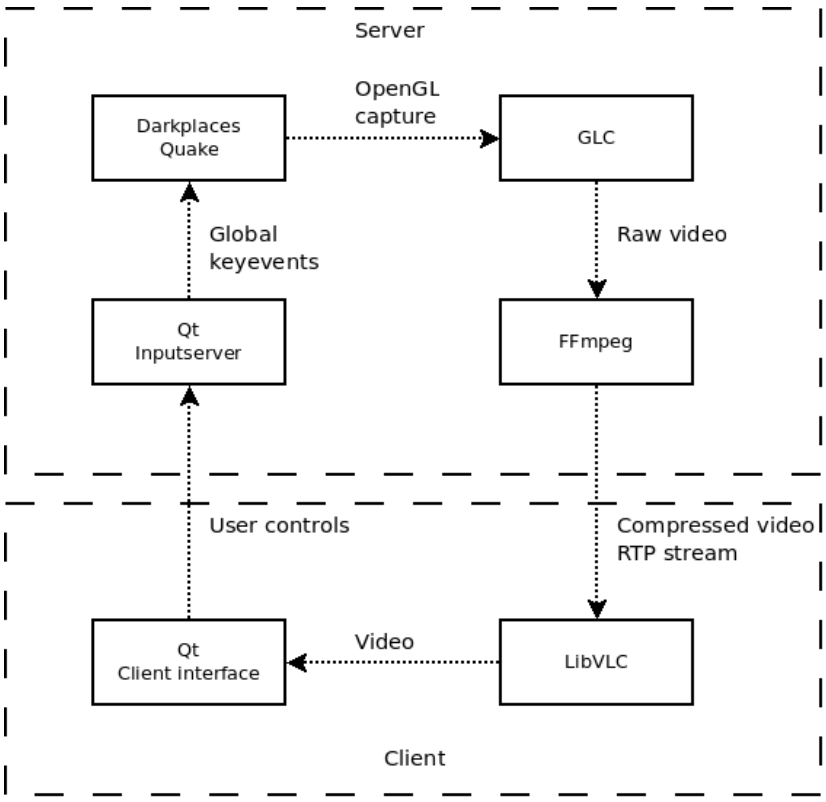
In this section, project design is presented with a help of UML graphs. Sequence diagram presents the data packages which are send over the network. Component diagram shows the third party and other software components in this project. Lastly, we introduce you the main algorithms and user interface in our program.

4.2 Sequence Diagram



In our sequence diagram, we have client computer on the left and server computer on the right. First, client sends a message to server to start the game. When server has received the message it starts to send video stream and is ready to receive user controls. User plays the game by getting video stream on his/her screen and sending controls over the network. When user wants to close the program client sends a message to server to close the connection. We use UDP packages to communication so client and server do not send ACK messages to each other.

4.3 Component Diagram



In our project we use third party software components quite a lot. We use FFmpeg, GLC, Qt and Darkplaces in our server computer. Darkplaces is the game which is played by the user from the client computer. GLC captures the video stream and FFmpeg packs the raw video and sends it to client. Client receives compressed video with LibVLC. Client interface and Inputserver is developed by using Qt framework.

4.4 Main Algorithms

Our main algorithms are mostly third party software components. We use open source program FFmpeg and h263+ video codec to capture, pack and stream the video. With FFmpeg video is captured and streamed. Video codec h263+ packs the video so we can use our network bandwidth more efficiently. RTP is a standardized protocol for sending audio and video packages over the Internet. Despite of specificied codecs, our design allows us to quickly change used

4.5 User Interface

User starts to use the program by executing the file Client.exe. First, program opens the menu screen (Picture 1) where user can press “Connect” to start playing the game or “Exit” to exit the program. After pressing the button “Connect” program connects to the server computer and starts playing the with the game user interface (Picture 2).

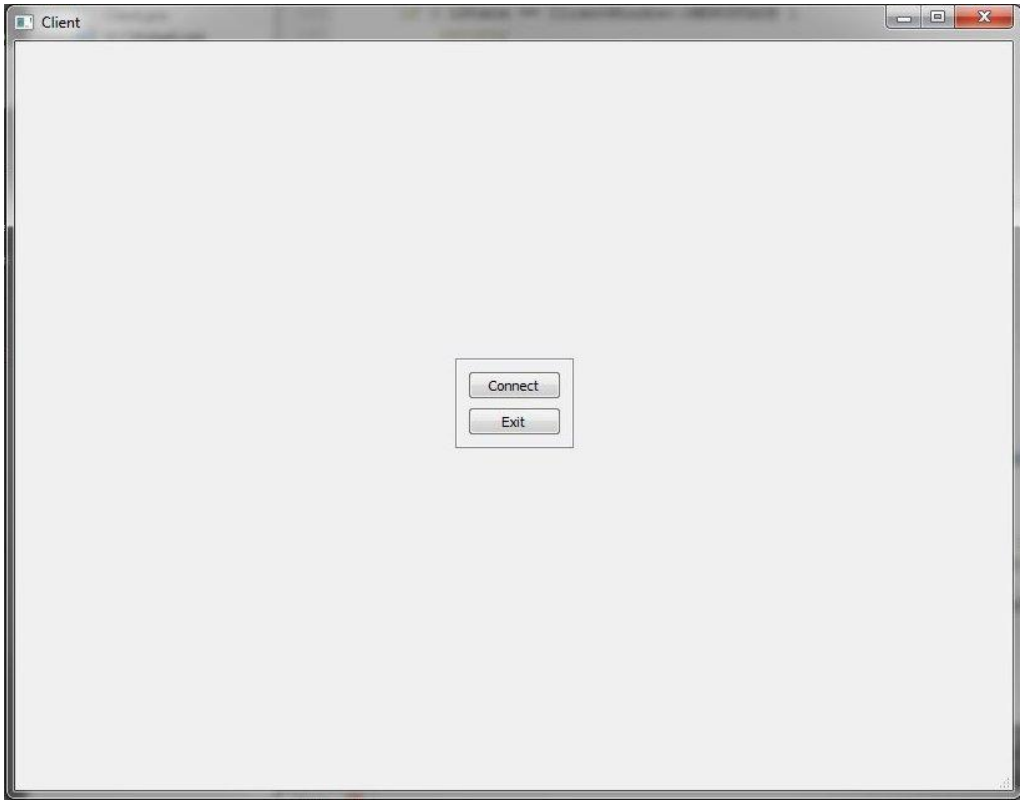


Image 1. The program menu


Image 2. The game stream

5 COMPILING & EXECUTING INSTRUCTIONS

5.1 Compiling Instructions

This part of document explains specific compiling instructions (if any).

- Laitetaan copy-pastella ku on jotain laitettavaa (Tuomas)

5.2 Executing Instructions

This part of document explains specific running instructions (if any).

6 REFERENCES

[1] Jaaksi A. & Aalto J-M. & Aalto A. & Vättö K. (1999) Tried & True Object Development, Industry-Proven Approaches with UML. Cambridge University Press, SIGS Books.