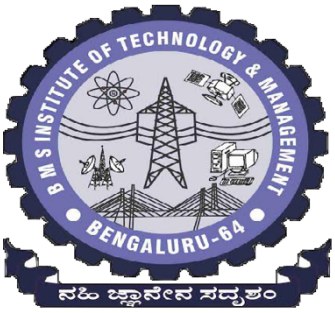
BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT

Yelahanka, Bangalore – 560 064

Department of Computer Science & Engineering



Synopsis for the Project work

"CLIENT SERVER SIMULATION USING OPENGL"

Submitted By:

Mohmmed Daaniyaal 1BY15CS051

Bhargav Sagiraju 1BY15CS016

Under the Guidance of

Mr.Shankar R

2017-2018

BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT
BANGALORE-560064

Department of Computer Science & Engineering

PBL Synopsis of COMPUTER GRAPHICS Mini Project - 2017-2018 (6th Semester)

Batch No:	Guide Name:	Submission Date:
CSE-A1 / A2	Mr. Shankar R	28-03-2018

Project Title
CLIENT SERVER SIMULATION USING OPENGL

Sl No	USN	Name
1	1BY15CS051	Mohammed Daaniyaal
2	1BY15CS016	Bhargav Sagiraju

Signature of Guide

ABSTRACT :

In computer science, client-server is a software architecture model consisting of two parts, client systems and server systems, both communicating over a computer network or on the same computer. The client-server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. Often clients and servers communicate over a computer network on separate hardware, but both client and server may reside in the same system. The client process always initiates a connection to the server, while the server process always waits for requests from any client. A server host runs one or more server programs which share their resources with clients. A client does not share any of its resources, but requests a server's content or service function. Clients therefore initiate communication sessions with servers which await incoming requests. The client and server communicate with each other through a well-known application protocol.

INTRODUCTION :

In today's era of the internet majority of the communications happens through the client-server architecture model. All online transactions such as browsing the web, shopping on e-commerce portals, online banking transactions, etc are made possible through communication between the client i.e. the user of a service and the server. This process of communication is not as simple as explained above.

Clients and servers exchange messages in a request–response messaging pattern. The client initiates a request to the server and the server returns a response. This exchange of messages is an example of inter-process communication. To start the actual communication between the client and server, first a connection has to be established between the client and server through an interface called socket. Once the connection is set up, the client and server exchange messages through this socket. To communicate, the computers must have a common language, and they must follow rules so that both the client and the server know what to expect. The language and rules of communication are defined in a communications protocol.

The exchange of messages between the client and server is in the form of data packets . Bytes are grouped together to form a data packet. Apart from the actual message the data packets also contain additional information such as source and destination address, protocol and its version, error control information, etc.

MOTIVATION :

Simulation of any process helps in better understanding of it. This project will demonstrate the internal working of a client-server architecture model. It would help in visualizing how connection between a client and server is established. This system can be used for educational purposes to teach and explain the working of client and server and network programming.

EXISTING SYSTEM :

Several tools exist today for viewing communication such as client server simulations such as wireshark and ns2 which do have a well based Graphical User Interface (GUI) to show where packets are being sent. For example wireshark has a very limited system to display the headers and the packet being sent to analyze the information and the direction of the packets being sent.

The exact positioning of the nodes is unknown where the client cannot be differentiated from the client. So the system will be unable to exactly show where exactly the system will respond to the packet sent and how.

LIMITATIONS OF EXISTING SYSTEM :

The existing system of displaying a simulated system of client server simulations has always been flawed because of its inability to display a simulated GUI. It has always been unable to convey a convincing display about the packet transfer and the receiving party and also to know where the packets and data is being sent to the server.

The insight into the data transfer has been missed out thus defeating the purpose of the protocol analyzer tools such as wireshark. A layman or a common person does not have the necessary knowledge to understand where all his packets are being transferred to and where exactly there is an issue.

Tools like ns2 cannot provide a very detailed and accurate analysis of the necessary data packets sent and received. The varied details and actions taking place in a simulated system are very limited.

PROPOSED SYSTEM :

Our system covers the existing in the way that only the necessary and essential transfers are noted and also the actual details of the packets being sent are noted. The details about what is in the packets is not noted down and only the simulations of the working packets is clearly shown. This will help a person understand what is happening in a network without understanding the technical details in the network.

SYSTEM REQUIREMENTS AND SPECIFICATIONS :

- ## FUNCTIONAL REQUIREMENTS :

Functional requirements are the necessary requirements for functioning of a software or a system. Functional requirements required for this system are –

1. **Processor :** Intel Core i3 or higher.
2. **Memory :** 1GB or more.
3. **Operating System :** Windows / Linux / MacOS X.
4. **IDE :** Code Blocks.
5. **Compiler :** GNU / GCC Compiler.
6. **Graphics Library :** OpenGL.

Various methods such as **establish_connecton, send_data, disconnect_connection,** etc are used for the implementataion of the system.

- ## NON – FUNCTIONAL REQUIREMENTS :

Non – Functional requirements support the functioning of the system. They have a check on the entire system as a whole. Non – functional requirements of the system are -

1. **Reliability :** How reliable is the system to deliver the system its service.
2. **Safety :** The ability of the system to perform without failure.
3. **Availability :** The ability of the system to render its services at all times whenever requested by the user.
4. **Dependability :** This is a property of a software / system that establishes a confidence about the system to the user.
5. **Security :** The system is secure such that no outside attacker can intercept the system.

PROPOSED METHODOLOGY :

Our system uses the OpenGL library to simulate client server communications to show what exactly is happening in the foreground and background. This will help all the people working on the network to see what exactly is happening inside the network.

We use several shapes to show what is happening in the network. To represent the packets in the network where the client and server are communicating. The OpenGL shapes exactly show where and how the shapes move.

Packets can be shown as to how they move and where they end up at. This is a unique way to understand the packets in a network.

REFERENCES :

1. https://www.wikipedia.org/
2. https://stackoverflow.com/
3. https://www.opengl.org/
4. Donald Hearn & Pauline Baker: Computer Graphics with OpenGL Version, 3rd Edition, Pearson Education,201.
5. Edward Angel: Interactive Computer Graphics- A Top Down approach with OpenGL, 5th edition Pearson Education, 2008.