ASSIGNMENT NO. 08

TITLE: MINI PROJECT

AIM: To create a Distributed Application for Interactive Multiplayer Games.

TITLE OF GAME:- name of game

OBJECTIVE:
Students should be able to understand:-

- Distributed Application
- Distributed Architecture
- Description of Game
- Architecture for Game
- Game Design

PROBLEM STATEMENT:-

Student should project statement of their game.

TOOLS / ENVIRONMENT:

- S/W:
  - Can be developed on any platform Windows /Linux.
  - C/C++/Java
  - Student can add tools used for implementation like netbeans, eclipse, VS Code, etc
- H/W:
  - Any basic configuration loaded machine (e.g. P IV)

THEORY:

Distributed Application

A distributed application consists of one or more local or remote clients that communicate with one or more servers on several machines linked through a network. With this type of application, business operations can be conducted from any geographical location. For example, a corporation may distribute the following types of operations across a large region, or even across international boundaries:

- Forecasting sales
- Ordering supplies
- Manufacturing, shipping, and billing for goods
- Updating corporate databases

State of the art telecommunications and data networks are making distributed operations of this sort increasingly common. Applications developed to implement this type of strategy allow businesses to reduce costs and enhance their offerings of services to customers around the world.

Distributed Architecture

1. Client-Server Architecture

Before we get into exactly what different tier systems that are involved with client-server architecture, first we need to understand. What does it entail to create client server architecture for a Java application? The simplest way to have the design of a client server architecture, is to have framework that is segmented into different tasks or workloads that are provided by either a service or a request that are either the server or the service requester (client).

Hence, the name client-server architecture where you have the client who on one side is creating some request in which the server acts as the function of the requests and completes the task. In this scenario, the client is facilitating all communication with the server side therefore the client side does not share any reasons with other clients.

2. Broker Pattern Architecture

So, what is a broker pattern? Basically, a broker pattern application is a distributed system using decoupled components that interact with each other using remote service calls. What makes broker patterns unique is the broker component, which coordinates the communication of the decoupled components. When a client requests a task in an application, the broker receives that task or service and in turn redirects the task to the appropriate server to execute the task.

The main way achieve this design is to create what is called a CORBA. CORBA is a standard practice that is defined by the object management group (OMG) where they help facilitate the communications of technologies for these complex systems.

3. Service-Oriented Architecture

SOA is a design strategy in which applications use services that are available in the network. Each of those services can be formed together to create an application. SOA does have necessary requirements of the following:

- Standardized service contract
- Loose coupling
- Abstraction
- Reusability
- Autonomy
- Discoverability
- Compositability

The requirements above describe the components that make up a SOA. But there are also different roles defined by a SOA application. Those roles can range but in general most are three roles of the service provider, service consumer/requester and service broker.

The roles are pretty self-explanatory but the service provider acts as producer of the information to the service registry, the service broker helps to ensure that the information from the service provider is available to the service consumer. Basically, the responsibilities

become a hybrid of both structures where the client-server now has a slightly decoupled service that acts as the broker.

Description of Game

Student should write here description games including:-

- Problem Definition
- Motivation of Game
- Game elements
- Game Requirements

Architecture for Game

Student should draw Architecture for Game and explain components of Architecture.

Game Design

Student should write detail steps in designing of game.
Draw flowchart of game.

IMPLEMENTATION:

(Students should write here the implementation for their program. Students should attach printout of their programs with commands used to run the programs. Also attach the proper outputs of programs.)

CONLCUSION:

In this assignment we learn to use distributed architecture to design a Distributed Application for Interactive Multiplayer Games.

REFERENCES:

- "**Distributed Systems: Concepts and Design**" by George Coulouris, J Dollimore and Tim Kindberg, Pearson Education, ISBN: 9789332575226, 5th Edition, 2017
- **"Distributed Systems"**, Maarten van Steen, Andrew S. T, Third edition Version

FAQ:

1. What is a distributed application?
2. How do distributed application works?
3. What are the major challenges with distributed applications?
4. Which types of distributed application architecture you select for game design? And why?
5. Other than game, explain any one example of distributed application?