DYNAMIC LOAD BALANCING FOR MASSIVELY MULTIPLAYER ONLINE GAMES

By

SARMAD ABDULMAGED ABDULAZEEZ

A thesis submitted in partial fulfilment of the requirements of Liverpool John Moores University for the degree of Doctor of Philosophy

October 2017

DECLARATION

I, Sarmad Abdulazeez, hereby declare that the work in this thesis is original and has been completed by myself, except where reference is made to other works, and has not been submitted for any examination at this university or any other learning institutions.

Sarmad Abdulazeez

ABSTRACT

In recent years, there has been an important growth of online gaming. Today's Massively Multiplayer Online Games (MMOGs) can contain millions of synchronous players scattered across the world and participating with each other within a single shared game. Traditional Client/Server architectures of MMOGs exhibit different problems in scalability, reliability, and latency, as well as the cost of adding new servers when demand is too high. P2P architecture provides considerable support for scalability of MMOGs. It also achieves good response times by supporting direct connections between players.

This thesis proposes a novel hybrid Peer-to-Peer architecture for MMOGs and a new dynamic load balancing for massively multiplayer online games (MMOGs) based this hybrid Peer-to-Peer architecture. We have divided the game world space into several regions. Each region in the game world space is controlled and managed by using both a super-peer and a clone-super-peer. The region's super-peer is responsible for distributing the game update among the players inside the region, as well as managing the game communications between the players. However, the clone-super-peer is responsible for controlling the players' migration from one region to another, in addition to be the super-peer of the region when the super-peer leaves the game.

In this thesis, we have designed and simulated a static and dynamic Area of Interest Management (AoIM) for MMOGs based on both architectures hybrid P2P and client-server with the possibility of players to move from one region to another. In this thesis also, we have designed and evaluated the static and dynamic load balancing for MMOGs based on hybrid P2P architecture. We have used OPNET Modeler 18.0 to simulate and evaluate the proposed system, especially standard applications, custom applications, TDMA and RX Group. Our dynamic load balancer is responsible for distributing the load among the regions in the game world space. The position of the load balancer is located between the game server and the regions.

The results, following extensive experiments, show that low delay and higher traffic communication can be achieved using both of hybrid P2P architecture, static and dynamic AoIM, dynamic load balancing for MMOGs based on hybrid P2P system.

ACKNOWLEDGMENTS

In the name of Allah, the Most Gracious and the Most Merciful. I am extremely thankful to Almighty ALLAH for his blessings and providing me with the ability to perform this research, without which none of my work would have been done.

My sincerest thanks and profound appreciation goes to my supervisor professor Abdennour El Rhalibi for his guidance throughout my PhD study, without which this thesis could not have been produced in the perfect form. His insightful comments, constructive criticism and suggestions have given me invaluable guidance in different stages of my research. He has been very kindly, helpful and supportive for all these years. I greatly appreciate his care and concern throughout my PhD study.

My gratitude and appreciation also go to my second supervisor, Professor Dhiya Al-Jumeily, for his valuable comments, guidance and suggestions.

I take this opportunity to express my gratitude to my mother-in-law (May God have mercy on her) for her support during my research.

I would also like to express my gratitude to my parents for their support, advice and encouragement during my PhD study.

I will be eternally grateful for the endless love and affection bestowed upon me from my beloved wife for her endless patience, continuous encouragement, and support.

LI ST OF PUBLICATIONS

During the period of my research, the following works have been published:

- Abdulazeez Sarmad, El Rhalibi Abdennour, Al-Jumeily Dhiya.2017. Dynamic Area of Interest Management for Massively Multiplayer Online Games Using OPNET. Tenth International Conference on Developments in eSystems Engineering (DeSE'2017). France, Paris.
- Abdulazeez Sarmad, El Rhalibi Abdennour, Al-Jumeily Dhiya.2016. Simulation of Area of Interest Management for Massively Multiplayer Online Games Using OPNET. Ninth International Conference on Developments in eSystems Engineering (DeSE'2016). Liverpool, UK 31st August-1 st September 2016.
- Abdulazeez Sarmad, El Rhalibi Abdennour, Al-Jumeily Dhiya. 2014. Simulation of Massively Multiplayer Online Games Communication Using OPNET Custom Application. 5th International Workshop on Digital Entertainment, Networked Virtual Environments, and Creative Technology (DENVECT'16) Messina, Italy June 27 - 30, 2016, Co-Located with IEEE Symposium on Computers and Communications (ISCC 2016).
- Abdulazeez S, El Rhalibi A, Al-Jumeily D. 2016. Evaluation of Scalability and Communication in MMOGs. 13th IEEE CCNC 2016 (IEEE Consumer Communications and Networking Conference) 9-12 January 2016, Las Vegas, Nevada, USA.
- Abdulazeez SA, El Rhalibi A, Merabti M, Al-Jumeily D. 2015. Survey of solutions for Peer-to-Peer MMOGs. 2015 International Conference on Computing, Networking and Communications, ICNC 2015, :1106-1110. Anaheim, California, USA.
- Abdulazeez SA, El Rhalibi AE, Al-Jumeily DA. 2014. Load Balancing Techniques for Massively Multiplayer Online Games: A Survey. PGNet2014. Liverpool John Moores University, Liverpool, UK.

CONTENTS

List of Figures

List of Tables

1

INTRODUCTION

1.1 Massively Multiplayer Online Games (MMOGs)

MMOGs differentiate themselves from other online games by allowing hundreds of thousands of players to share a single game world, and allow for players to move and interact with each other in the virtual world. These give the potential for participants to socialise through online games, which gives a major advantage to MMOGs over games with a single player. Nowadays, multiplayer online games have become more popular in the game industry. MMOG is one of the most common types of online games which characteristics such as massive numbers of players in a single game world.

Generally, MMOG is define as one of the most common of computer game that enables thousands of players to participate concurrently in a shared game world when they are connected via an Internet. The number of players when the word "massively" implied in the 1990s were just around 100 simultaneous players. Nowadays, it has deteriorated to become a significant problem because the number of players has increased to hundreds of thousands [1]. MMOGs are the game that able to support hundreds or thousands of players using the Internet. Number of players in real games typically follow a power law distribution [2]. The number of players in EverQuest game are typically around 2,500 players concurrently[2].

MMOGs can be divided into various genres of games types such as Massively Multiplayer Online Role Playing Games (MMORPGs) for example World of Warcraft and most of the existing commercial MMOGs are MMORPGs, Massively Multi-player Online Real Time Strategy (MMORTS) for example Prime World [3], and Massively Multiplayer Online First Person Shooter (MMOFPS) for example Planetside [4]. The first MMOGs appeared in the late 1970s and they were called Multi-User Dungeons (MUDs) [5]. MUDs are text-based adventure games that execute a fantasy world as a collection of "rooms". MUDs were very

simple games due to the limitation of the bandwidth and computational power at that time. MUDs provide an easy user interface that present its player with a description of the present room that the player is in, in addition to objects, other players, or non-player characters in the vicinity. All the actions are performed by typing corresponding commands to run them. Actions involve activities like, killing monsters, moving among rooms, completing quests, and chatting with other players, and so on. The first MUDs games were pretty simple applications based on a client-server system [1]. They were running on university mainframes and bulletin boards. Figure 1.1 shows the screenshot of an old MUD game called Dungeon Crawl Stone Soup [6].

Figure 1.1: Dungeon Crawl Stone Soup Game

Nowadays, MMOGs still use client-server architectures to run games like the first MUDs, however, more powerful server clusters are used, which are much faster than the old servers.

During the 1980s and 1990s with the development of computer technologies, MMOGs start using graphical user interface with more possibilities/functionalities, but there was no significant evolution of MMOGs until the end of 1990s,when Ultima Online , Lineage and Everquest brought the MMOG genre to the mainstream market [1].

Modern MMOGs introduce 3D graphical representations and utilise a massive virtual world to be explored within the game. Modern MMOGs also can support many more players who inhabit, communicate, cooperate and compete with each other on a wide scale in a shared world that provides unprecedented immersive gaming experiences with convincing audio and

visual content. With the increase of subscriber numbers, MMOGs have become one of the most significant commercial game sectors. Therefore, the MMOG industry is considered to have become one of the most successful, promising online businesses, and has demonstrated an amazing profit potential.

The essential premise in most MMOGs is that the player assumes the role of an avatar in a virtual world. Modern MMOGs have developed greatly when compared with traditional games and they are supported for advanced 3D graphical representations of virtual worlds and player avatars.

1.2 Thesis Statement

One of the key characteristics related to the massively multiplayer online games (MMOGs) is the large number of players, having hundreds of thousands of participants concurrently. In addition to the rapidly increasing number of players with interaction between them in the game world space, the interaction of players and the migration from one region to another generates a high level of traffic and can cause overload on the support network. This issue is gradually growing according to the number of players inside the game world. In order to distribute the load between the regions in the game world space in a good manner, load balancing technique was introduce to deal with this issue. In this thesis we investigate the issue of dynamic load balancing technique for massively multiplayer online games (MMOGs) based on hybrid P2P architecture. We investigate dynamic Area of Interest Management (AoIM) for MMOGs based on hybrid P2P system. We also investigate the scalability of the system with the increasing number of players within the same game world.

The biggest challenge when using hybrid P2P architecture for MMOGs instead of traditional client-server architecture is to deploy the players among the regions in the game world space. Consequently, the use of super-peer and clone-super-peer to manage and control the region to provides an easy way to maintain the region as well as to update the game to each player inside the region and consequently, the use of dynamic area of interest management in order to reduce the overall delay and traffic received of the network. Therefore unlike previous load balancing systems, in this thesis we propose a novel dynamic load balancing system for MMOGs based on hybrid P2P architecture in order to obtain a better level of scalability, AoIM, reduce the overall delay, and reduce the overall traffic received.

1.3 Research Problem

MMOGs have diverse workload features compared with traditional game applications. Load balancing technique related to the mechanism to distribute the load of processing that occurred when the peers join and leave the system. The load balancing problem becomes more important when the number of MMOGs players in rapidly increased. The problem of balance and distribute the load of MMOGs based on hybrid P2P architecture are most important and challenging areas of research. The major problems in this research are to perform an efficient and effective load balancing technique for MMOGs based on hybrid P2P architecture.

1.4 Aims and Objectives

This research project aims to investigate the possibility and applicability of designing a new dynamic load balancer for the MMOGs based on hybrid P2P architecture, as well as, designing and simulating a new dynamic Area of Interest Management for MMOGs based on hybrid P2P architecture. The aims of such a system are to provide more scalability of MMOGs based on hybrid P2P system, dynamic AoIM and dynamic load balancing taking into consideration the migration of players from one region to another inside the game world space; in addition to reduce the overall delay and traffic received in order to provide a good level of consistency, efficiency, responsiveness, and low latency of the system. The following are the objectives of this thesis:

- We investigate how MMOGs based on hybrid P2P system differ from MMOGs based on client-server system. In particular, scalability, performance, responsiveness, and distribution.
- We propose the design of MMOGs architecture based on both hybrid P2P and client-server architectures and modelling, evaluating, and simulating by using the standard application of OPNET to ensure the scalability and responsiveness.
- We also investigate the use of custom application of OPNET Modeler to design MMOGs to ensure better game communication between the players as well as to manage and control the game in an easy manner.
- We investigate the use of OPNET Modeler to design and simulate static and dynamic area of interest management (AoIM) for MMOGs as well as the ability for the players to move and migrate from one region to another or even in the same region of the game world space.

- We investigate the suitability of the proposed system over different numbers of players and various numbers of regions.
- We evaluate the performance of the proposed system considering the building of new games and the possibility of the dynamic load balancer for distributing the load among several regions, as well as, considering the players' migration from one region to another.

1.5 Contributions

The contributions of this thesis are highlighted in the following:

- **Design and simulation of hybrid P2P architecture for MMOGs.**

  The general concept of the hybrid P2P architecture for MMOGs is to provide a new mechanism to distribute the players among the regions in the game world space. This architecture used the concept of both super-peer and clone-super-peer to control and maintain the region. This architecture allows the server to work efficiently because the super-peer and clone-super-peer will work to reduce the load of the game server. The architecture is designed to be simple, flexible, and easy to apply for different structural topologies, with the ability to change the number of player inside the regions.

- **A novel and extensible dynamic Area of Interest Management (AoIM) for Massively Multiplayer Online Game.**

  The designing and simulation area of interest management for MMOGs based on both architectures hybrid P2P and client server. The possibility for creating a static and dynamic group of receiver inside the region in the game world space in order to reduce the delay, response time, latency, traffic received, and provide good level of consistency, performance of the system.

- **A novel Static and Dynamic Load Balancing for Massively Multiplayer Online Games.**

  We design and simulate a static and dynamic load balancing for MMOGs. Our dynamic load balancing algorithm is applying for the MMOGs based on hybrid P2P system, it has the possibility to distribute the load dynamically among the regions, it supports a large number of players that play concurrently in the same game world, and it also supports gaming interactivity and players' migration from one region to another during the simulation. Though dynamic load balancing is

primarily designed to support the distribution of players' load between the regions in MMOGs based on hybrid P2P architecture.

- **Evaluation of the system using OPNET Modeler**
  OPNET Modeler 18.0 have been used to evaluate the proposed system. We have used standard applications of OPNET to evaluate the scalability of our hybrid P2P architecture. However, OPNET custom applications have been used to design a game application in order to evaluate the proposed architecture and ensure the scalability, consistency, responsiveness and performance. In addition, we have used TDMA, RXGROUP to design and evaluate a dynamic AoIM and dynamic load balancing in order to provide a dynamic aspect to the player nodes in the game world.

1.6 Structure of the Thesis

This thesis is organised in eight chapters, each chapter addressing different element of the investigation. Chapter 1 introduces the research problem along with the aims and objectives of this study. It also identifies the research scope and describes the structure of this thesis. Chapter 2 gives an overviews of some topics related to the research area. Chapter 3 reviews the literature to investigate recent studies related to our research area. Chapter 4 introduces an evaluation of the scalability of MMOGs using OPNET Modeler standard applications. Chapter 5 discusses the design and simulation of a custom application for MMOGs based on hybrid P2P and client-server architecture. Chapter 6 presents the design and simulation of a static and dynamic area of interest management for MMOGs. Chapter 7 presents the design and simulation of a static and dynamic load balancing for MMOGs. Chapter 8 introduces the conclusion and the future work of the research.

2

RESEARCH BACKGROUND

2.1 Introduction

This chapter presents massively multiplayer online games in more detail and highlights the characteristics and concepts related to this area. In addition, it presents the architectural approaches commonly used to develop MMOGs platform. These architectures include client-server, Peer-to-Peer, and Hybrid Peer-to-Peer architecture. The chapter begins with an overview of MMOGs from an academic research perspective.

2.2 Massively Multiplayer Online Games (MMOGs) Development

2.2.1 Academic Perspective for MMOGs

Academic research for MMOGs has become increasingly popular in the last few years. The research networking concepts are related not just to computer game technologies but also other domains such as network simulations, distributed systems and serious games. MMOGs are a nontrivial distributed network application type [7]. MMOGs have a variety of complex features that deserve in-depth study. For instance, they require near real-time interactions, reliability, scalability, consistency, security, authentication, robustness, efficiency and persistent data storage. In addition, they involve many challenges and benefits for the development of a considerable number of non-game applications. MMOGs provide a cheap and safe training environment. For example, military, flight, medical surgery simulations in virtual world, distance e-learning, e-commerce, and manufacturing systems.

Both gaming environments and simulations can facilitate students' research of both special domain concepts and knowledge, as well as various skills such as pattern recognition, solving problems and decision making. Use of simulations in research can lead to a deep understanding of systems behaviour. Simulations work effectively when the system is impossible to

implement in the real world as well as studying the behaviour of a system over time [8]. For instance, the using of simulation will help to understand the traffic behaviour of network, as well as the great understanding of the network protocols that are used in the simulation. Nowadays, the most common use of multiplayer virtual environments is games. These immersive environments use real contexts, activities, and assessment [8]. The using of immersive multiplayer virtual environments is to let players participate in new worlds, inhabiting roles that would be inaccessible to them. Another significant element of online games is the great community that will develop around them to be able to share in these communities, problem definition groups emerge and solving problem occurs, and also a great deal of socialising. The next section will explain the essential concept of MMOGs.

2.2.2 MMOGs Concepts

There are general concepts that must be taken into account when designing a multiplayer online game whether using client-server or P2P architectures. Figure 2.1 depicts an overview of different components of a typical multiplayer game structure. Here, we discuss the general concepts and execution patterns used in most multiplayer online games.

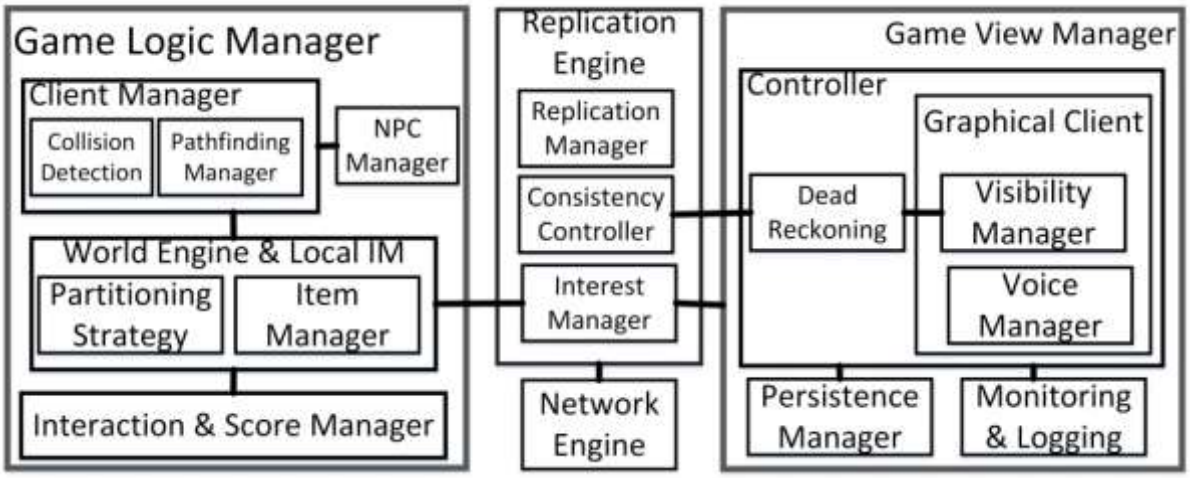


Figure 2. 1: Different Components of a Multiplayer Game [7]

2.2.3 Object Types

A typical multiplayer game world is usually made up of four types of objects [9]. The first type is immutable objects, for example landscape (the terrain) information. These objects are commonly designed and created offline and never modified during the game. Graphic elements for the landscape are usually installed as a static part of the game client software, however they can be updated using the software update techniques. The second type is characters (avatars).

These types of object are controlled by the player using an input device. States of avatar must be persistent and can be carried along from one login session to another. The avatar has three kinds of actions in the game world: player updates, player-to-object interaction, and player-to-player interaction. The third type is mutable objects like weapons, food, and tools that can be changed. The last type of objects is Non-Player Characters (NPCs) or so-called bots. These types of object can be characters or avatars that are not controlled by a player but are commonly controlled with AI. Figure 2.2 shows the object types in a typical MMOGs structure.

Figure 2. 2: Game objects and interactions

2.2.4 Player Interactions

There are typically three kinds of player interactions in the game world: player updates, player-to-object interactions, and player-to-player interactions [7, 14]. Player updates are interactions with the game world that just impact on the player himself. Player updates examples are position updates and graphical updates to the player's avatar. Player-to-object interactions are the interactions between a player and mutable objects in the game world. For instance, when the player drinks from a bottle, the state of the bottle object would change from full to empty, as well as reduce the thirst parameter of the player object. Player-to-player interactions are interactions among players in the game world. For example, fighting and trading, just influence on the states, such as life points of the players involved. However, player interactions with NPCs are typically based on the design of the game, and can be considered either player-to-object or player-to-player interactions. These types of interaction are more important when coping with consistency issues that arise from synchronous and discordant updates to the same object.

2.2.5 Object Replication

A player receives an instance of the game world when he/she joins a game. An instance sometimes can be a limited view of the game world, which is made up of different kinds of game objects. Most game engines use a primary-copy replication technique. There exists an authoritative copy especially for every object and avatar, called the primary or master copy. The secondary copies are called replicas. Each player stores on his computer copies of game objects which are of interest to the player. The update of object is performed first on the primary copy. The primary copies may always reside on the server or sometimes are held by clients, whilst primary and secondary copies are distributed according to the game architecture. If a player wants to perform an update on an object for which he does not have the primary-copy, he has to send the update to the primary copy. The holder of the primary copy decides whether to accept the update or not, and then sends the updated object to everyone that has a secondary copy, where the changes are applied. The update dissemination technique is totally similar to publish-subscribe methods that have been introduced in [10]. Each replica becomes a subscriber to the object's primary copy and receives publications (updates) from the primary copy. The basic elements of MMOGs will be described in the next section.

2.3 The Basic Elements of MMOGs

There are many basic elements related to MMOGs. The most important elements of MMOGs are a persistent game world, player avatars, and non-player characters. These elements are explained in details in the next sections.

2.3.1 Persistent Game World

One of the most important features in MMOGs is a persistent virtual game world or sometimes called a Persistent World (PW) [1]. It refers to storing all the players' profiles and inventories between login sessions. The persistency term comes from the ability to maintain and develop the state of the gaming world around the clock. In another words, a game world is always available to use by players at any time. Completely contrary to other types of games, the plot and events of a persistent world game continue to develop even while some of the players are not playing their characters. All the real world is persistent.

2.3.2 Player Avatars

MMOGs involves a virtual environment that is represented by the combination of entities. One of the most important entities is avatars, which represent the players in the virtual world.

They can be controlled in the virtual environment and interact with other entities. Precisely, a player may be represented by two types of avatars. The first type is called a pilot avatar that is present on the player's own machine. However, the second type is called a drone avatar which imitates the pilot avatar on all the other machines connected to the game world. A pilot avatar is controlled immediately by the user in the communication layer. However, a drone avatar is required to update state and behaviour to be delivered over the network, which may introduce a latency issue. Communication latency is considered one of the issues that affect the player avatar. It is defined as the amount of time required to transfer data from one place to another, as well as, referring to the network delay when two players are interacting in MMOGs.

MMOG designers often include a latency tolerance of 250 ms into the communication system by using different methods to assess the information about other players in the game, for example the new position of player avatars can be predicted according to its previously specified position, velocity, course and elapsed time. The most widely used approach which deals with these issues is dead reckoning [11].

2.3.3 Non-Player Characters

The latest class of entities is called Non Playing Characters (NPCs), which are characters in the virtual environment which are controlled by the intelligent part of the game (the AI integrated part of the game) rather than by the player [12]. Figure 2.3 [13] shows the typical type of NPCs in the Star Wars Online game. Every player must maintain a local copy of the game state to provide a shared sense of space among players in the game world. Often, when one of the players implements an action, the game state of all other players influenced by the action must be updated directly by real-time gaming events. If the network caused transmission delays of events, it will result in inconsistent game states at various players' machines. Anubisath Sentinels [14] game is an example of real-time interactions, in which the states of massively multiple player online games must be tightly simultaneous. Traditionally, an interaction between two PCs in the game world is called a Player-vs-Player (PvP) interaction, however an interaction between PCs and NPCs is called a Player-vs-Non-Player (PvN) interaction.

Figure 2. 3: NPCs of Star Wars Game.

2.4 Online Computer Games Networking

Online Computer games are distributed real-time multi user applications, providing a virtual interactive world. There are typically three categories for the distributed interactive real-time: Networked Virtual Environments (NVEs), Military Simulations, and Multiplayer Computer Games (MCGs) [15]. It is requisite to understand the related aspects of networking which impact on MCGs, in order to understand the complexities of networked computer games. MMOGs are a special subset field of MCGs. MMOGs have been developed for the majority of gaming types, the most popular type are Massively Multiplayer Role Playing Games (MMORPG).

2.4.1 Resources of Networking

There are three resource limitations that are faced for distributed simulations when designing and performing solutions for MCGs: network bandwidth, network latency and computational power. These resource limitations are described in the following section:

- **Bandwidth**

The players in the MMOGs environment need to constantly receive the updates of the game state from the game server to update the game status. These updates are more important to improve the progress of the game virtual world, as well as for interaction between players. The updates of the game state need adequate network resource, in order to ensure the high quality and scalability of a game. Bandwidth refers to the transmission capacity of a communication line such as a network. In other words, bandwidth refers to the amount of data that can be

carried by a digital communication medium per unit of time. The requirement of bandwidth for MMOGs can be calculated depending on the average message size, update rate, and number of recipients (active players). MMOGs with millions of (active) subscribers have high bandwidth requirements because of their dynamic environment or high update rates. [5, 17]. The bandwidth of LAN ranges between 10Mbps and 10Gbps, whilst bandwidth range of WAN between tens of kbps in dial-up modems up to 1.5 Mbps in T1 and 44.7 Mbps in T3. Broadband internet connections are usually related to as WANs which are most typically used for consumer-based MMOGs and have a drastic reduction in bandwidth. However, LANs have a fixed size but they support a restricted number of users, whereas WANs allow global connections. Bandwidth requirements depend on the number and distribution of users, as well as the size of messages transmitted[15].

- **Latency**

MMOGs require fast responsiveness to user's actions due to MMOGs being real-time game applications. In the context of MMOGs, the responsiveness is defined as the time it takes for the player to implement the actions in the virtual world. However, latency refers to the length of time that takes the message to transfer from source to the destination. Latency depends on the architectural design, networking delays, processing time, as well as on the number of messages transmitted between players in the game world. Nowadays, there are different kinds of massively multiplayer online games. In Real Time Strategy (RTS) and Role Playing Games (RPGs) [18] the players are focusing on game strategy instead of responsiveness. The player confirms the avatar to make a probably complex and lengthy permanent action, for instance, go to a destination, and the avatar executes the requested action. The latency influences the playability of the game and thus user satisfaction, when the latency is higher than the tolerance threshold of the game.

- **Computational Power**

Dealing with network traffic requires additional computing power to be used when running a distributed system. This is usually the least restrictive factor due to the processing speeds and memory capacity of the current generation of desktop computers and controllers. However, a multi-processor server solution dealing with 100,000 entities (very possible in some architectures) produces a CPU load of about 80% [15]. There are some techniques used to reduce network resources which will be explained in the next section.

2.4.2 Techniques to Reduce Network Resources limitations

There are several techniques used to reduce the network resources limitations. These techniques are explained in the next section.

- **Packet Compression Technique limitations**

The main objective of compression technique is to reduce the number of bits needed to represent particular information used to communicate state changes within the virtual world between the players and the server, as well as between players with each other. Thus, the packet compression of the network can offer an intuitive approach to reduce network traffic by reducing the amount of data transmitted, and save bandwidth at the cost of computational power. There are two kinds of compression according to the ability to preserve information content. The first type is lossless compression technique which preserves all information and the reconstructed data is exactly the same as the data before compression. However, the second type is lossy compression technique. This technique works by leaving out less relevant information so that the deformation in the reconstructed data stays unnoticeable [19]. Furthermore, there are two kinds of compression techniques related to data in packet format. The first kind is called internal compression and focuses on the information content of a single independent packet without reference to the other packages that were transmitted previously. The other kind of compression techniques are called external compression, which use information that has been already transmitted and supposed to be available to all recipients. External compression can be considered a better technique for large amounts of data at a time, and, thus, it can better observe repetition in the information flow.

- **Packet Aggregation Technique**

This technique works to reduce requirements of bandwidth by merging several packets into one large packet and transmitting it, and this leads to reduce the overhead caused from packet headers. The saving of bandwidth can be considerable, based on the original size of data in the packets, the size of packet headers, and the actual number of merged packets.

- **Dead Reckoning**

Dead reckoning is used to minimise bandwidth consuming by sending update messages less frequently and estimating the state information between the updates. The state update can contain extra information for predicting how the state will change in the future. Dead reckoning uses an approximation method to predict the player's state such as the next position in the game

world. The message can include some information for example the velocity, which is used to predict the change of data over time. Dead reckoning is an integral part of the Distributed Interactive Simulation (DIS) and High-Level Architecture (HLA) standards [20]. Dead reckoning can be used in client-server or P2P architectures and the operation is the same [21].

- **Area of Interest Management**

Providing relevant information to each player in the game world is an efficient method for consistency management. The aim of using AoIM is decreasing the number of transmitted messages by specifying the potentially interested receivers [15]. This allows communicating the minimum amount of information that a peer needs to interact with other peers in the game world, in order to display an accurate state of the world to the players. This idea is not only related to P2P MMOGs, but has also been used in client/server architecture for virtual environment [22] and in distributed environment [23]. However, interest management becomes an inherent part of the network organisation, when used in conjunction with P2P infrastructures. This method is used with the aim to organise the connections between players in the game world so that they are only contacting with players who possess information pertinent to themselves.

- **Load Balancing**

The global load balancing approach takes into account load information from all servers and processes all nodes to determine the optimal solution to divide the game scene. One of the main objective of using load balancing technique is to optimize resource use, reduce the throughput, reduce the response time, and avoid overload of any single resource.

2.5 MMOGs Architectures

There are three main architectures used to develop MMOGs. These architectures are client-server, Peer-to-Peer, and hybrid Peer-to-Peer, which are explained in the next section.

2.5.1 Client-Server Architecture

The client-server model is a traditional method of designing networked multiplayer games, due to the simplicity and efficacy of the approach, with many examples of its usage. It consists of one or more authoritative game state servers, and one or more clients. The servers make changes to the game state that are propagated to the connected clients that in most cases (for cheat prevention and game fairness issues) cannot directly modify the state, but instead inform

the server of their input for their particular player on the server. The client-server mechanism has been used as a traditional model for writing both simple networked multiplayer games and applications, and remains the simplest mechanism of consolidating the game state to ensure correct state synchronization.

The load balancing can be achieved effectively by distributing the processing of the game across several servers. A widely used load balancing technique in MMOGs is deploying players to different servers (or so-called player-based) when the player joins the game. Client-Server architectures are widely used due to the fact they are comparatively simple to implement, control and maintain the state of the game. In addition, scalability can be done by increasing the number of servers, the processing resources (such as the power of CPU, and memory) and the bandwidth through some techniques, for instance clustering. This solution is inefficient and inflexible due to the increased cost of dedicated hardware required to provide good scalability [7].

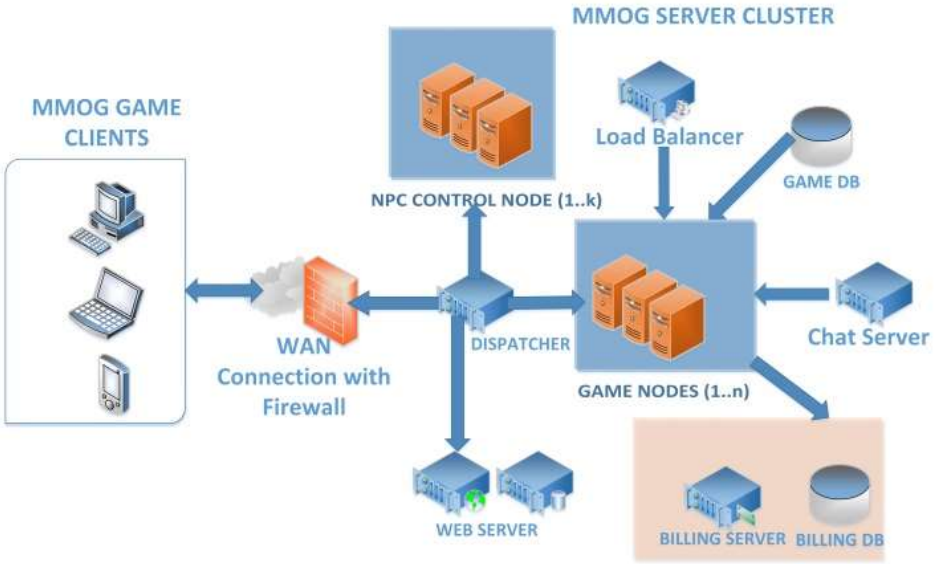


Figure 2. 4: MMOGs Architecture based on Client-Server [7]

Figure 2.4, illustrates a common MMOGs system architecture based on server cluster, involving game nodes, game database, and chat server, billing server, web server, NPC control node, dispatcher and load balancer. This system architecture is centralised as all the clients communicate directly with one of the servers, which synchronises the game state updates and communicates them to the clients. This is different from MMOGs Peer-to-Peer in which clients can communicate directly among them. This is introduced in more detail in the following section. The key disadvantage of using client-server architecture is scalability. Even the best-

provisioned servers are unable to handle more than a few thousand players because of the limitations resources of the server [7].

2.5.2 Peer-to-Peer Architecture

Peer-to-peer (P2P) system refers to a distributed system and every peer in the network takes equal status and control. Recently, P2P networks have been growing faster and also become more widespread in the applications of the Internet all over world. Peer-to-peer networks allow game designers to transfer a big part of the game processing and the load of bandwidth to their participant's peers. In other words, the infrastructure of P2P authorises sharing the resources of a computer, such as CPU, bandwidth, and memory storage without the necessity of centralised servers [24]. The main advantage of P2P architecture is better realised in file-sharing applications, self-organising, being adaptable and providing highly scalable systems. A P2P network is presented as an overlay network on top of a physical network. The responsibility of overlay is for storing and locating services. The overlay has the ability to provide a peer with the address of another peer that has a copy of the desirable content. P2P architectures can be broadly categorised into three key types: Centralised P2P architectures, decentralised P2P architectures, and hybrid P2P architectures [25]. Centralised P2P systems employ one or more servers in the P2P network to let peers determine the desired contents. Hence, the responsibility of the central servers is to maintain the addresses of peers, as well as the contents or services they provide. The main disadvantage of centralised P2P systems is a single point of failure of the server, as well as the problem of server bottleneck. One of the key advantages of this network is there is no single point of failure or bottleneck problem in the overlay system. However, one of the main disadvantage of using P2P architectures is security. Cheating is easier in a P2P system [2]. Another drawback of using P2P architecture is the difficulty to manage, and control the system because there is no central server that maintains the overall system.

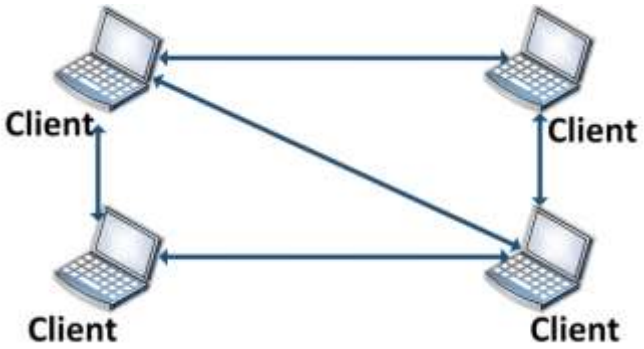


Figure 2. 5: Typical P2P Architecture

2.5.3 Hybrid Peer-to-Peer Architecture

The main idea of the hybrid P2P architecture is to combine the benefits of both client-server and peer-to-peer architectures [7]. Hybrid P2P overlays provide a dynamic server concept for exploiting all the features of centralised and decentralized systems. The essential goal is to provide improved scalability compared to client-server system, lower cost of game distribution, control of the game state, and deployment of the game update over the players in an easy way. In the hybrid P2P system, the centralised server is responsible for maintaining the game state and keeping it persistent and consistent, as well as, doing the main administration operations, for example authorisation, authentication, and content distribution. These operations require much less resource consumption than the operations done by the server in the client server system. Therefore, the hardware cost for these servers or adding new server would be substantially reduced [26]. The hybrid P2P architecture also has several challenges to overcome. One of the most important challenges is the capability of the system for adding new peers and allocating them to one of the regional servers in the network system. Therefore, the system will need an effective way to distribute the work load between the regional servers, being able to re-distribute the load when one of the regional servers becomes unavailable. Figure 2.6 shows the hybrid P2P architecture. The hybrid P2P system can be divided into three categories based on what is handled by the P2P system [7].

- *Cooperative Message Dissemination*: The game state is maintained by one server or multiple servers depending on the server-based architecture. However, it uses a P2P approach to update game dissemination. Usually, players send their interactions and queries directly to the server. After execution, the server uses a P2P multicasting technique for sending the updates to all players. This mechanism reduces bandwidth requirements as well as providing cost savings at the server side.

- *State Distribution:* The game state is distributed between peers. Peers are responsible for the execution of player actions because they hold primary copies of objects, but, all or part of the communication among peers might be managed by one server or more. Furthermore, the server is responsible for centralised processes such as authentication, and control of the joining and leaving of players. This mechanism achieves good scalability by distributing the cost of state execution among peers.
- *Basic Server Control:* P2P overlay is responsible for both message dissemination and state distribution. The essential role of the servers is to keep highly sensitive data, such as players' logins, payment information, as well as players' progress and game state. The responsibility of the servers is to control the joining and leaving of players. However, servers do not preserve games state or execute state dissemination.

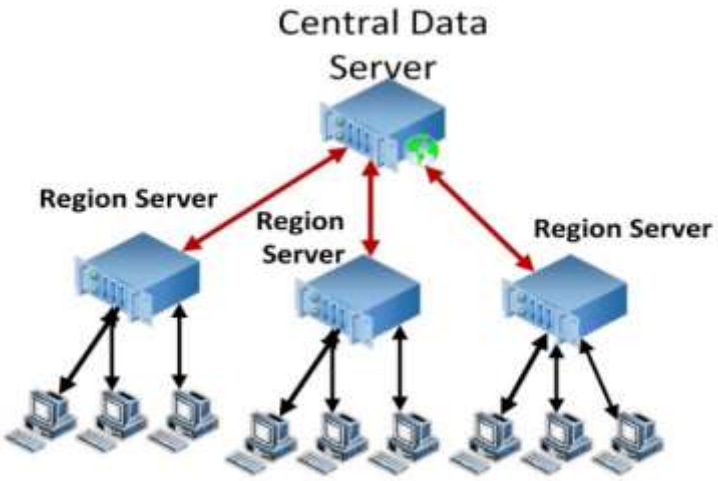


Figure 2. 6: Typical Hybrid P2P Architecture

Table 2. 1: Comparison of Different Architectures

Architecture	Advantages	Disadvantages
Client-Server	Simplicity Easy management Consistency	Scalability Cost Single point of failure
Peer-to-Peer	Scalability Fault tolerance Cost	Security Harder to develop Consistency
Hybrid Peer-to-Peer	Scalability Easy to develop Security Cost Fault tolerance	Single point of failure

2.5.4 Unstructured P2P Overlay Networks

Early P2P systems used an unstructured technique since they did not follow any special structure when building the network. Therefore, there was no connection between the stored contents and peers in the overlay. In fact, the research is based on random walk and flooding mechanisms, it is therefore difficult to expect the location of the queried service. The overlay networks organize peers in an arbitrary graph in a flat or hierarchical style. In other words, peers are not connected depending on a predefined deterministic scheme. Furthermore, it is never confirmed that a data item will be retrieved, even if that data item exists in the network. Thus, unstructured overlays are not considered appropriate as a basis for MMOGs based on P2P system, where all data items must be available at all times [7]. An unstructured network works well for content sharing and Voice over Internet Protocol (VoIP) and include networks, for instance Gnutella [27], Freenet [28], BitTorrent [29]. The methods for searching in this category are random walks, flooding, expanding-ring and Time-To-Live (TTL). Unstructured P2P overlay networks were originally designed to support quick information dissemination and content-based searching in extremely dynamic distributed environments. In unstructured P2P networks, there is no guarantee that a data item which exists in the network can be found by a search without searching the full network. These problems are solved by introducing structure in the organisation of the nodes in the overlay. Structured P2P overlay networks will be introduced in the next section.

2.5.5 Structured P2P Overlay Networks

Structured P2P networks [30] have been invented to process the shortcomings of unstructured systems. Structured P2P networks were introduced to provide effective search methods that guarantee to access the destination within a small number of hops. The overlay network works by assigning keys to data items and arranging its peers into a graph that maps each data key to a peer. An overlay network can be described as a virtual logical layer, which sits on top of the physical layer provided by the internet. By using the given ID, the structured graph can achieve effective discovery of data items. Structured P2P networks oblige a specific connection structure between peers. The standard in this type of networks is Distributed Hash Tables (DHT) [31]. They connect peers based on their ID to enable effective ID-based routing between the peers in the network. DHT is capable of

balancing object hosting and the load of query while transparently reconfiguring after node failures.

One of the most commonly used overlay networks is Pastry [32]. It is a self-organising overlay network of nodes, peers are ordered by using a logical ring structure for the identifying key space, as shown in figure 2.7. Each node in the Pastry peer-to-peer overlay network is assigned uniformly constructed unique 128-bit peer identifier (peerId). The other types of overlay networks in this category are Content Addressable Network (CAN) [22], Tapestry [23], Chord [33], Kademlia [34], and Viceroy [35].

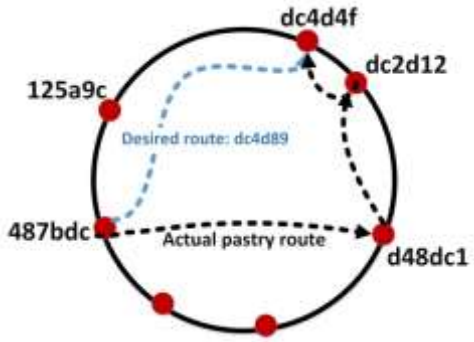


Figure 2. 7: Pastry - DHT Structure [43]

Many researchers have used these P2P overlay networks approaches to propose solutions for MMOGs. Iimura et al. [36] proposed the zoned federation model, which deals with MOG based on peer-to-peer networks and maintains a DHT of peers, which are mapped to regions of the game world. In this approach, the DHT is used to structure the regions while the game events are distributed through the region peers by immediate connections to the participating players within the region [36]. Knutsson et al., [9] proposed SimMUD using Pastry and Scribe to divide the virtual world into regions with fixed size to provide AoI. Each region is mapped using the DHT, to a node coordinator. Lu Fan et al. presented Mediator by employing a structured DHT using Pastry and Scribe. Mediator utilizes Quadrant zoning to adapt a geographic AoIM technique. Peers are arranged into a hierarchical super-peer network through the DHT [37].

2.6 Issues within P2P and Online Games

P2P networking for MMOGs has become a progressively prevalent area of research. However, there has been no significant commercial utilisation of this approach. Whilst the reasons for this lack of adoption are complex, two issues which are central to this are:

The proposed P2P architectures have yet to address the full range of issues in relation to scalability, security, availability; these are significant research challenges that must be addressed in order to see a fully-realisable P2P MMOG. Current implementations are typically applied in simulated environments which oversimplify the problem domain and may not accurately reflect their industrial counterparts. Other issues are related to online games development, where it is necessary to appreciate the aspects of networking which affect their performance. When designing and implementing solutions for online games, software architects are faced with three physical resource limitations, which need to be balanced in order to deliver a smooth, coherent online experience: Network Bandwidth: This is the transmission capacity of a communication line. Network Latency: This is the length of time that is incurred when a message is transported from one node to another. Computational Power: Handling network traffic requires additional computational power. Related issues and solutions are discussed below highlighting state consistency, game event dissemination, and game state persistency [38].

2.6.1 State Consistency

Consistency in MMOGs means that each player has uniform information, regardless of the underlying architecture or implementation. In other words, it indicates the ability of all players to agree on the current state and view in the game world. To achieve great consistency, the architecture must guarantee that processes running on remote peers are tightly coupled. This typically needs high bandwidth and low latency. Inconsistent conduct happens because of the implementation parallel and conflicting updates, and consistency techniques have to avoid or correct the current state. For instance, when two players shoot a third player at the same time, all players should see the organise updates and just the first shot must be successful at all replicas. Another example, when a player drinks a healing potion to raise his points of health, and just depending on the increased amount of health, is able to pick up a weapon. These actions must be executed in the same order at all replicas in the game world; otherwise inconsistencies might become more visible. Furthermore, other kinds of inconsistency are caused by updates loss. Most games use UDP messaging protocol in order to cope with latency issues, but the message loss may be possible. To deal with this problem, a reliable TCP protocol must be used for sending updates messages [39].

MMOGs based on P2P architecture, this problem is solved depending on authority distribution within the game world. However, if some messages are received out of order, then

execution can lead to an inconsistent state when the out of order messages are causally dependent.

2.6.2 Event Dissemination

A large number of players in MMOGs connect over the Internet to join a virtual world. There is an imminent need to make the virtual game worlds concurrent for all players, as tightly as possible to ensure user satisfaction. The importance of information concerning game events for a special player is constituted by the virtual world vicinity. When it is connected to the real world, close events are more important than remote events. Therefore, game event dissemination is concerned with how relevant information is indeed delivered to the players. However, in the consistency concept, it is designed to determine how events and updates are distributed in the network. The option of a game event dissemination technique is broadly determined by the underlying interest management mechanisms used in MMOGs based on P2P [7].

2.6.3 Game State Persistency

MMOGs is indicated as a Persistent World and it often runs for years, due to the information and inventories of player's avatar and objects being maintained when the player temporarily leaves and joins again the game another time, as well as any changes they have influenced on the world. When a player returns to the game, the player retrieves its former state information and continues to play. State Persistency is a significant issue in MMOGs based on P2P gaming. The data of the game is distributed over the players' computers; if there is not state persistency, some chunk of game state will be missed when a player shuts down his/her computer, which could lead to the disrupting of the game.

2.7 Network Simulation Tools

2.7.1 OMNeT++ Simulation Environment

OMNeT++ is considered one of the open source component-based discrete event network simulators [2]. The simulator is used to support standard wired and wireless IP communication networks, but some extensions for Wireless Sensor Network exist. OMNeT++ is popular simulator, extensible and actively maintained by its user' community. OMNeT++ uses C++ language for simulation models. One of the important features in OPNET++ is the support of graphical tools for simulation building [2]. The disadvantage of using this simulator is limitation of available number of protocols.

2.7.2 NS-2 Simulation Environment

NS-2 is a commonly used general purpose discrete event simulation tool for sensor networks [40]. NS-2 is currently used in academic research because it is easy to extend and based on open source. NS-2 simulations are used with C++/C (OTCL) languages to model the network. One of the main drawback of NS-2 is the scalability. As well as the simulator lacks an application model [41].

2.7.3 OPNET Modeler Simulation Environment

OPNET[42] (Optimized Network Engineering Tool) is a powerful simulation network that enables the simulation of a large-scale network. In other words, the OPNET modeller is a developing environment appropriate for the design, simulation and analysis of networks. Due to that OPNET contains a large number of Ethernet, ATM (Asynchronous Transfer Mode), IP (Internet Protocol), and TCP (Transmission Control Protocol), as well as the easy way for simulating and flexible analysis of all parts of the data network. We have used OPNET Modeler to evaluate our system for several reason. The first reason is the simulation works with Finite State Machines (FSM). The second reason it provides a graphical editor interface that is used to build models for different network entities from physical layer to application processes. There are three graphical editor used in OPNET Modeler, the network editors, node editor, and process editor as shown in the figure below.

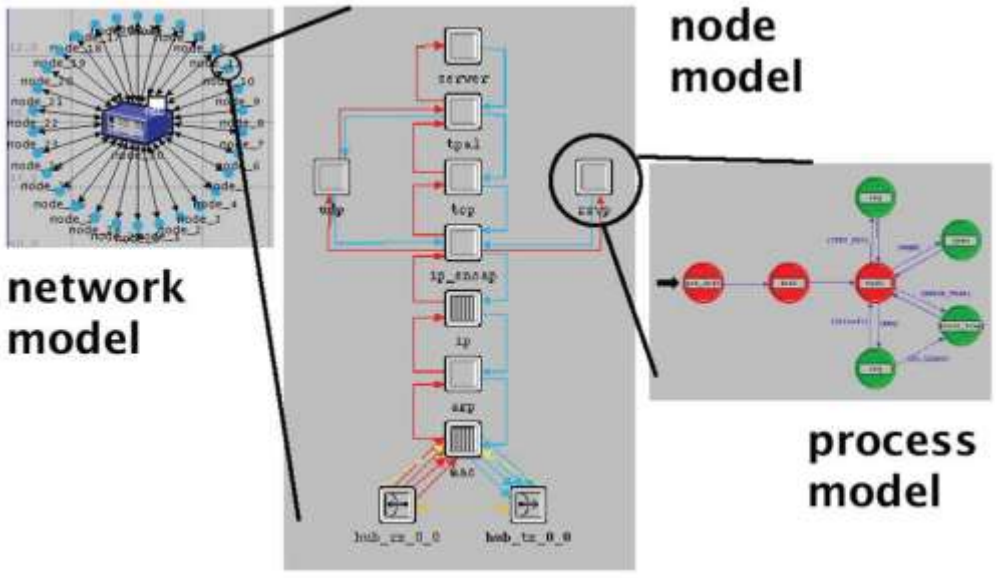


Figure 2. 8: OPNET Graphic Editors (Network, Node, and Process Models)

The third reason is the possibility for interconnecting between OPNET Modeler and external system. The interconnection is called co-simulation, using External System Interface (esys) to connect between the OPNET Modeler and the external system. This interface is used to ensure the control and data exchange between each system. The fourth reason is the development environment for the simulation and analysis of networks from various fields and various levels of complexity. OPNET is used to model devices, protocols, and behaviours with nearly 400 modelling functions. OPNET has a special feature to allow the user to modify the existing system and produce new ones and the flexibility of creating several scenarios in the same project to create a good design for the network. Using Proto-C language to program the OPNET model [43].

2.8 Conclusion

In this chapter, we have discussed the characteristics and concepts related to the research area. We have explained the development of MMOGs, and the basic elements of MMOGs. In addition, we have presented the main three architectures that are commonly used to design the MMOGs, and explained the advantages and disadvantages of each architecture, as well as reviewed the issues within P2P and Online Games. In addition, we have explained different types of simulation environments and the reasons for choosing the OPNET Modeler to design and evaluate our proposed system.

3

LITERATURE REVIEW

3.1 Introduction

Massively Multiplayer Online Games (MMOGs) are considered some of the most popular kinds of online games. Traditionally, MMOGs have used client/server architecture. This architecture has several advantages such as a single authority ordering events, a central data repository, as well as being easy to control and secure. However, there are major disadvantages when the number of players is increased, such as increased latency time, server bottleneck, limited storage capacity, as well as limited computational power by the game server. There are many researches which have given different ideas to solve these problems. We will explain these researches in the next sections.

3.2 MMOGs based on Client-Server Architectures Researches

The client-server architecture has been considered one of the predominant paradigms to implement traditional MMOGs. Bharambe et al. [44] have presented the design, implementation, and evaluation of Colyseus, a distributed architecture for online multiplayer games. Colyseus takes feature of a game's tolerance for weakly consistent state and predictable workload in order to meet the tight latency constraints for game-play as well as maintain the costs of scalable communication. Furthermore, Colyseus provides a good distributed query interface and efficient pre-fetching subsystem to assist locating and replicating game objects before they are accessed at a node. There are two kinds of game objects: immutable and mutable. The authors assume that immutable objects are globally replicated and each node in the system has a copy since they are updated extremely rarely. Immutable objects are such things as map geometry, game code, and graphics. There are three main components in Colyseus that are used to manage objects on each node. These components are State Partitioning, Execution Partitioning, and Object Location [44]. To evaluate the system, the

authors have implemented Colyseus and modified Quake II, a popular first person shooter game. The results of Quake II and Colyseus-based games with hundreds of players illustrate that Colyseus efficiently distributes game traffic among the participating nodes, as well as allowing Colyseus to support low-latency game-play in order to support more players than existing single server designs, with similar per-node bandwidth costs [44]. There are some limitations in this research. One of the most important limitations is the limited number of players supported by the game world. Also this system is implemented just for the FPS games, not for all types of MMOGs games.

Alecu [45] presents an architecture for MMOGs based on the client- server system that allows the player the mechanism for interacting with any other online player. The author proposed four tier architecture to support web-based games that can separate the client requests based on frequency. Also, allowing for a very robust system with few active servers that is eligible for supporting tens of thousands of online players. The author also proposed a novel solution that preserves the convergence between two players based on their session information and predicting future actions for those players in order to reduce the amount of data that needs to be transferred between servers when two players stored on various machines want to interact [45]. Based on small-scale and large-scale tests, the proposed solution made a considerable improvement by reducing the overall data that needed to be transferred by 12% [45]. The research did not consider the server overhead problem when the number of players is rapidly increased.

Distributed architecture for Massively Multiplayer Online Role-Playing Games [46]. Dividing the game world into regions, each region is controlled and managed by several servers. Clients can be clustered together based on players' locations in the game world. The client has just one main point to contact the server at all times, in order to send actions to the server. For all clients, the server responsible for acting as the main point of contact is determined by the location of the player in the virtual world. The client carries on sending actions to the same server, unless it is notified by another server. This architecture proposes to decrease the bandwidth requirements for both clients and game servers, address consistency, hotspot, congestion and server failure problems typically found in MMORPG, and allows seamless interaction among players residing on areas handled by several servers [46]. The main problem in this research is the cost of adding a new server to the game world when the number of players is increased.

3.3 MMOGs based on Peer-to-Peer Architectures Researches

P2P architectures have the highest possibilities for scalability as each peer that joins the game will add new resources to the network system. A special advantage of adding new resources to the system is that it will not add any extra cost for the game provider. Furthermore, all the responsibility is distributed across several nodes, each of them carrying only a bit of the load. The failure of any individual peer will not influence any other players in the game world. There are many researches which have worked to develop the P2P network system, some of them related to MMOGs.

Merabti and El Rhalibi [47] discussed several issues related to MMOGs, the lack of using a simulation environment in order to study and evaluate network protocol and architecture. P2P based architecture has been used to provide a good level of scalability, fully distributed, fault tolerant for MMOGs. The authors have managed the design and implementation of a modular MMOG called 'Time-Prisoners', by using a P2P system developed in Java and JXTA. The use of P2P overlays offers the opportunity to dynamically organise in clearer way for the users. Players are grouped depending on their locations in the virtual world, and P2P overlay allow the design of scalable techniques for distributing the game state to the players as well as maintaining the consistent world [47]. This research did not provide a high level of scalability compared to the huge number of players in MMOGs.

GauthierDickey et al. [48] developed a fully distributed peer-to-peer architecture for MMOs. Players in the game world can send messages straight to each other, thereby this mechanism will reduce delay and eliminate localised congestion. To create storage capacity, the author used the combination of the storage resources of all players that can easily override a single server. Furthermore, the capability to implement thousands of remote processes in parallel by using user machines provides unparalleled computational power for MMOs. The author divided the architecture into four major components. These components are authentication, communications, storage, and computation. The authentication component is responsible for controlling access to the game. The communication component locates the method of sending messages between players. The storage component provides long-term storage for the world state. However, the computation component works to schedule computations across the player base [48]. This research provides a good scalability, however there are some problems such as no component for game update.

Hampel et al. [49] proposed to combine MMOGs with a Peer-to-Peer network. The authors proposed a game architecture eligible for exploiting the flexibility and scalability of P2P networks, as well as developing an MMOG architecture based upon the DHT overlay network by using Pastry in combination with the event distribution through Scribe. The game world is divided into many regions. These regions can be selected randomly and they can be connected to provide a large game world environment. Each region is managed by using a region controller, a peer that has been selected to control and maintain the game states of all other peers in the same region consistent [49].

Chan et al. [50] presented a method to support MMOGs based on a P2P overlays system called Hydra. Hydra tries to provide a simple increased server-client programming model. In order to implement a group of protocols to support the required interface, the authors have designed scalable techniques to distribute the game state among the players, as well as to maintain consistency in the face of node failures. The resulting system dynamically scales with the number of online players. It is more resilient and has a lower deployment cost compared to the centralised games servers. The primary results show that Hydra imposes just a small message overhead and is thus scalable [50]. One of the significant problems in this system is the security. There is no cheat prevention for the Hydra architecture.

Knutsson et al. [9] presented a method to support massively multi-player games based on a peer-to-peer overlays system called SimMud. This architecture takes advantage of the fact that players in MMGs display area of interest, and subsequently can be formed as self-organizing groups based on their locations in the game virtual world. In order to create a system that dynamically scales up and down with the number of players. The authors have designed scalable methods to distribute the game state among the participating players and maintaining consistency in confronting node failures. The results of the proposed system were dynamically scaled with the number of online players up to 4,000 to prove the possibility of this approach. Also it is more flexible and has a lower deployment cost compared with centralised games servers [9]. This architecture has no centralised authority system to control and manage the game state update.

Bauer et al. [51] proposed a methodology mechanism to evaluate the scalability of several architectures in order to specify the most convenient one for specific game types. The proposed system is very general in that it supported centralised, distributed, and hybrid architectures. It is implemented to the client-server, peer-to-peer and novel federated peer-to-

peer architecture. A quantitative model has been presented to evaluate the scalability of different game architectures. Defining cost functions to determine the amount of processing and networking resources that a given architecture has required. The effect of different game types can be obtained by assigning the parameters connected with the functions cost to convenient values. The authors applied these cost functions to different architecture such as the client-server, peer-to-peer and federated peer-to-peer architectures, and discuss the trade-offs among them for a set of parameter values [51]. This architecture is scalable but does not have a clear mechanism about the joining and leaving of a game, as well as the effect of players' failure on the game state update.

Hu et al. proposed VON [52] Voronoi based Overlay Network. It is a simple and effective design P2P network system in a fully distributed architecture that provides low-latency, and a message-efficient manner. In VON architecture, each node is represented as a site in the Voronoi diagram. Also, defining AOI neighbours as the nodes whose locations are within its AOI. There are two types of neighbours Enclosing neighbours and boundary neighbours. Enclosing neighbours are nodes whose regions directly surround the given node, however boundary neighbours are AOI neighbours whose enclosing neighbours might partly lie outside the AOI. Each node maintains a Voronoi diagram for all AOI neighbours and directly connects them to reduce latency. Position updates are sent to all connected neighbours when a node moves, for example AOI neighbours beside any enclosing neighbours beyond AOI. Neighbour discovery is achieved by using notifications of boundary neighbours, as they know both the moving node and other nodes beyond the AOI, using simulations for both fixed and dynamic AOI to evaluate VON system. The simulations are implemented in more than 3,000 discrete time-steps, where nodes move randomly with a fixed speed. Simulation results illustrate that VON can principally provide more scalability than existing methods, by bounding the per node resource consumption, however achieving high structure consistency and reliability [52]. This system does not provide any mechanism to deal with peer failure, as well as the solving of the hot spot problem, when all players like to play in the same area inside the game world space.

Jeppesen and Monnike [53] developed the concept of a dynamic graph based on P2P network system such as central node, nodes, and supernodes. The central node is considered as the network entry point and network founder, while, nodes represent participating players in the game world. However, Supernodes are responsible for handling communication between nodes in the game world. These Supernodes are located at a particular location in the game

world, which is not predefined. Supernodes work dynamically according to where they are needed to balance the network-load. The results illustrate that the proposed architecture provides good performance among players in the vicinity in terms of in-game location, as well as providing low latency among interacting players, which leads to a smooth and responsive gameplay [53].

Wang et al. [54] proposed a fully distributed P2P communications architecture for Network Virtual Environments called FDPCA. The capability of a node is fundamentally calculated depending on the node's CPU, memory and bandwidth. This system calculated optimal AoI-radii for each node in the game world in order to deal with node crowding packet loss problems. To construct the multicast tree, the node's fan-out have been chosen depending on the node's degree in theory which can be calculated using node ability. Each node has number of children in the tree, and can be dynamically adjusted in the system, further the nodes can balance load of message autonomously based on node load. The simulation results illustrate that the proposed algorithm provides a significant improvement on multicast efficiency, as well as achieving good scalability [55]. The proposed architecture has the problem of security because there is no centralised server.

3.4 MMOGs based on Hybrid Peer-to-Peer Architectures Researches

A hybrid game architecture [56] that maintains centralised control of the game state, while dramatically decreasing server bandwidth. The architecture is used to combine client-server and peer-to-peer event distribution, so that just extremely important events are processed by the game server. Also the architecture uses measurements and controlling to make sure that each player is able to handle event distribution and is indeed providing this service. Hybrid architecture allows game companies to support more synchronous players while still providing a controlled game experience, in order to dispose of the cheating problem. The experimental results for the hybrid architecture indicate to the considerable saving in bandwidth for the central game server. Peers acting as regional servers are responsible for handling most of the bandwidth required for game state updating. Also, results are especially applicable to role-playing games, since the majority of moves in these games are always positional [56]. There are some limitations in this research, one of the most important limitations is the scalability. This hybrid architecture is not scalable for the increasing number of MMOGs. Likewise, the architecture allows the players only positional movement (just inside the region).

Hybrid MMOG Server Architecture (HYMS) [57] proposed to decrease outgoing bandwidth consumption from MMOG servers. This architecture distributes some functions from servers to the chosen clients in the game world, and those clients are responsible for event notification to other clients in order to decrease the outgoing traffic from servers. The clients communicate with server functions based on a peer-to-peer system. However, the central server maintains the whole game. Also, the client-server relation is maintained between players and the game server to handle consistency, failure and security problems. Furthermore, using redundant cell-daemons scheme for unexpected client crash. The results showed that up to 80% of server-side outgoing traffic can be eliminated even if only 10% of clients are cell-daemonable [57]. The limitations of this work are the game server still handles the whole game and is responsible for updating the game state for all the players in the game world, also this work needs to simulate the proposed architecture to prove the results closer to the real world.

Hybrid architecture for distributed virtual environments [58] that utilizes both client-server and peer-to-peer architecture design. It provides a higher scalability than existing architectures at a lower cost to the publisher. The user is responsible for supporting a small group of interactions, however the powerful servers will be responsible for dealing with the critical services such as user authentication, game persistence, and high-density user interactions. Peer-to-peer system communication and peer services will cost the game publishers nothing. This architecture works to distribute computation and, most importantly, network traffic to the user systems, and therefore game publishers can support a large number of players at very low cost, eventually decreasing the price of releasing and maintaining and control for MMOG [58]. This research moved the most common services to the peer machine, some players' machines do not have sufficient resources to handle these services, and therefore it will be a burden for the user, and it causes many problems such as game inefficiency.

Mirrored Arbiter Architecture [59] A Network Architecture for Large Scale Multiplayer Games that combines Client-Multi-Server (CMS) with Peer-to-Peer with Central Arbiter (PP-CA). This architecture utilises all the benefits of PP-CA, however resolves the major problems in PP-CA by using interest management technique and multicast. Players in the game world are divided into groups according to interest management technique. Each group of players is handled by using an arbiter. The arbiter is responsible for managing and controlling a global game region state and maintaining the consistency issue. When the arbiter receives an update from a player that is inconsistent with its game region state, it will work to ignore the update and send the correct region state to all players in the group. However, if any

players did not receive the correct region state message, thus those players will still have an inconsistency problem, and they will send back the conflict update to the arbiter again. The arbiter is responsible for revealing that conflict update and sending the correct region state to all clients again until the inconsistency problem is solved. To evaluate the performance of the proposed MA architecture, the authors implemented a multiplayer game called "TankWar". Both analysis and experiment behaved to show the efficiency of MA architecture [59]. This architecture is not scalable for the increasing number of MMOGs. Also, the author only focused on the performance of the MA architecture.

3.5 Area of Interest Management (AoIM) Researches

The game world of MMOGs contains an abundance of information, however in fact a single player needs just a small part of that information. The interest management is considered the best way for specifying useful information relevant to each player in the game world. Therefore, providing relevant information to each player in the game world is an efficient method for consistency management. There are several researches which have been published to provide good AoIM methods. These researches are discussed in the next section.

Ahmed and Shirmohammadi [60] introduced a dynamic AoIM for MMOG. The authors proposed to scheme AoIs to the virtual space instead of mapping the virtual space to the AoI, leading to a zone-less organisation of the virtual space. This zoneless aspect of MMOG is the result of a dynamic AoI that compensates the necessity of inter-AoI communication. The communication model combines a geometric algorithm in this response. Thus, the introduced concept decreased the communication load of the server to a considerable range not only because of its P2P nature but also for the confined area of interest. Furthermore, the AoI maintenance cost is considerably reduced by allocating the maintenance responsibility to a subgroup of players for each AoI. Due to the integration of the P2P communication model to the system, the scalability has improved. The model is evaluated and justified through proper simulation. The simulation results showed that the proposed approach can be a suitable solution for MMOGs based on P2P [60]. However, this research did not mention any mechanism for players' joining and leaving the game, which is an important mechanism in dynamic AoIM. However, Yu and Vuong [61] proposed fully distributed P2P infrastructure to support Networked Virtual Environment (NVE) applications, including MMOG. This architecture is a hybrid scheme focusing on NVEs' interest management. This scheme exploits advantages for both the DHT overlay and the unstructured P2P system providing a more scalable and fault

tolerant system compared to the existing methods, in addition to consistent performance with respect to neighbour discovery. In this work [61] the game map is divided into hexagonal zones, and each zone has an identical home node via the DHT mapping. Each cell has one master node. The master node must be registered in the home node of the cell. The communications between the master nodes are considered as the backbone for the neighbourhood dissemination. The master nodes in each cell are responsible for updating the neighbourhood information for all the slave nodes. The hierarchical structure and the effective neighbourhood message distribution algorithm drastically reduce the network bandwidth. The authors executed a simple game simulation and a visualisation tool, in order to simulate and visualise the proposed infrastructure [61]. This research did not provide any result that shows the saving in network bandwidth, or the load balancing performance of the network.

In another research, Ahmed and Shirmohammadi [62] introduced an auxiliary Area of Interest Management in zonal MMOG that works based on the P2P system. The proposed system consists of two main overlays called general/primary AoIM (G-AoIM) and Synchronized AoIM S-AoIM. The concept of S-AoIM is used to give synchronization to zone crossing. However, the G-AoIM concept is used to provide general overlay services for the game world. Using the S-AoIM and G-AoIM has two main benefits. The first one is to provide the indispensable data desired for seamless zone crossing, a usual phenomenon in zonal MMOGs. The second benefit is to support synchronization and adjust the load of the zonal MMOGs accordingly [62]. This system did not provide any clear results to compare with the previous research in the same area.

Boulanger [63] developed a tile partitioning technique that divides the game world depending on the geography of the world. Using Delaunay triangulation to divide the game world into triangles of a maximum area size in order to achieve a partitioning appropriate for interest management. All the partitioned space information is stored in a neighbour graph used to preserve the geographical information and can be used to search a local space to a player. Also he introduced four IM methods that used our neighbour graph to implement visibility-based and reachability-based interest management. The proposed algorithm was implemented by using the Mammoth framework and using a movement trace to perform experiments that collected in an event in which 28 real human players were participating. Also the author performed the same experiments by using two randomly generated traces [63]. This technique is considered as a static method for AoIM.

Pan et al. [64] introduced a hybrid IM method for P2P Networked Virtual Environments (NVEs). This technique has been implemented according to cell-based mechanism to minimize AoI updates in the area-based mechanism, as well as to decrease its communication overhead. Precisely, each entity in the NVE updates its cell information with all other entities when it moves from one cell to another. Furthermore, each entity is capable of preserving the residing cells of all other entities. An entity uses this information to send its AoI updates just to the entities within its AoI. Therefore, the number of AoI updates will decrease. To evaluate the proposed system, a multiplayer game based on P2P scenario is simulated and compared with the two previous approaches. The performance results illustrated that the hybrid mechanism decreases the upload bandwidth consumption by more than 25.28 percent, minimizes the overhead ratio from more than 67.54 % to just 25.17 %, as well as allowing approximately 5,000 players to share the same game with today's network upload bandwidth [64]. The author did not mention the simulator name, mechanised to simulate the proposed system, and the way for players joining and leaving the game.

3.6 Online Games Communication

Game simulation is widely used to evaluate new architectures and protocols. The game communication is considered one of the most important aspects in the game simulation. Some researches tried to deal with the simulation of massively multiplayer online game communication. Mohorko et al. present an overview of some suitable simulation tools for research on network technologies and communication infrastructure. The authors also show different possibilities of using the advanced simulation frameworks for simulating the networks. This system is of limited use for OPNET. This limitation leads to difficulties to control and change the communication model or create a new custom application [65]. Shirmohammadi et al. propose a P2P communication architecture for game network to provide reliability for important messages and decreasing network congestion by acknowledging only key messages. The system proposed, called Hybrid Distributed Simulation Protocol (HDSP, supports both best-effort delivery, for frequently occurring messages, and reliable delivery for important or "key" messages. HDSP simply implements the standard UDP protocol [66]. Denault and Kienzle used simulation in the context of MMOGs. They use Mammoth to simulate MMOGs in order to evaluate performance measurements such as CPU usage, memory usage. The authors define five simulation setups within the Mammoth MMOG framework, four using a single computer and the others in a distributed setting. The limitations of this system are the use of TCP protocol only, and there is no way to clearly define the game communication

in the simulation [67]. Webb et al. develop an application layer Network Game Simulator (NGS) and successfully simulated Client/Server system, Region based, and Neighbour based architectures. The simulator is very flexible and can easily be extended to contain new features and architectures. The disadvantages of using this simulator are that there are no complex routing algorithms, and NGS does not provide support to simulate the network stack. In addition this simulator has been used to just perform preliminary evaluation [68]. All these researches do not use OPNET Modeler simulation to simulate the game communication or the custom game communication. The used of OPNET custom game communication [69] provides more flexible features to manage the communication and control the traffic flow between network devices in both architectures, and support both TCP and UDP protocols. The next section introduces some researches related to the load balancing for MMOGs communication in the client-server system.

3.7 Load balancing for MMOG based on client-server architecture Researches

Recently, there has been a new technical challenge emerging in the gaming industry which focused on the possibility of managing the resources of game servers for massively multiplayer online games (MMOGs).

Denault et al. [70] introduced a dynamic load balancing mechanism that takes into consideration both the load related to game actions in addition to the load incurred by interest management. In this research, hybrid techniques have been used to split the main tasks the game's logic has to perform. Firstly, interest management (IM) and secondly, state update dissemination. The main concept is to partition the game world into small triangles that take into account the world geometry such as walls. A cell consists of many triangles connected with each other within the virtual world. There are two factors considered in this technique: the load associated with performing game actions and the load incurred over interest management [70]. The load balancing is achieved in two ways. The first way is called cell load model. In this model, when players and objects are equally distributed across the game world, the servers have the same cell size. However, it is uncommon that objects are equally distributed; also, most players resort to the most interesting zones in the game world. Thus, servers holding the heavily populated tiles have to do more processing for IM. The IM for each player has to be determined, and a large number of players in the cell. Therefore, the calculating load of the server is based on two components: the number of players into server's cell and the number of objects and players the server has subscribed [6]. The second way is called cell load

distribution. When the cell skips the threshold value and become overloaded, it tries to move some of its tiles to a neighbouring cell. It is quite important to select the tiles to be transferred. To do this, the cell calculates a priority value as follows; if the tiles neighbours and all members of the same cell, the priority is low such as 0. Otherwise, the priority is calculated according to the number of edge hops between this tile and the nearest tile of priority 0 [70]. However, the limitations of this technique are the authors did not mention the obvious criteria to calculate the threshold value for each player, and this research is based on the client-server system and it need more servers to cope with the increasing number of players. This causes the high cost for adding a new server to the game world.

Bezerra et al. [71] proposed a new mechanism for dividing the game world space into small regions based on kd-tree and achieves the load balancing of servers by repeatedly adjusting the split coordinates stored in its nodes. The important benefits for using kd-tree are: making this partitioning to allow a fine granularity to distribute the load and the readjustment of the regions becomes simpler. The load balancing approach is based on two main criteria: first, considering the servers as heterogeneous. It means each server may have a different quantity of resources. Second, the loads of the servers are not related to the numbers of players, but to the amount of bandwidth required to send state update messages to them. Because the number of messages sent by the players to the server will be growing linearly with the increase of player numbers, the number of state update messages sent by the server may not be good due to lack of bandwidth in the server. In this approach, using kd-tree with two dimensions, each node in the tree represents a region of the space and the node stores a split coordinate. Each node has two children and represents a subdivision of the region represented by the parent node. One of them represents the sub-region but has the parent node representing the region before the split coordinate. The region of the space is represented by the leaf node, which stores the list of avatars who are currently in that region. Ultimately, every leaf node is connected to the server of the game. When a server is overloaded, it performs the load balancing by using the kd-tree to modify the split coordinates that define its region, in addition to reduce the amount of content managed by the server. Also, each node in the tree stores two values: capacity and load of the sub-tree. The calculation of the load and capacity of a non-leaf node are equal to sum of the load of its children. However, the leaf nodes have the same values of the server connected to each one of them. The calculation of the load is dependent on the way of distributing the players among the regions. The players are deployed between the servers according to the bandwidth for each server. However, this method is not optimal when the

number of messages transmitted between players is large. Using a brute-force method for calculating the loads by calculating the number of messages that should be obtained by each player by unit of time [71]. This architecture does not support the scalability, therefore this is not suitable for the MMOGs with high numbers of players.

Lu et al., [72] presented a load balancing technique based on clustered server to achieve scalability. Allowing servers to transfer player actions to each other, however the responsibility for processing players' actions remains with the server to which they are initially earmarked. The system consists of two main levels: the application level and the database level. Firstly, the player connects to the server cluster through a load balancer, and he/she is then linked to a particular server in the application level. Application level is dedicated to meet the runtime requirements of game play. Through the database level, an application server can have access to the virtual world constructs and players' statistics via the load balancer that exists between application level and the database level [72]. But, this method is ineffective because the load balancer has not enough resources to cope with the increase of player numbers. Also, the cost of providing a large number of servers is very high as well as the cost of maintenance.

3.8 Load balancing for MMOG based on Peer-to-Peer architecture Researches

Usually zone size in MMOGs is fixed, but there are some techniques that cope with dynamic region size using for instance Voronoi diagrams. Most of these techniques do not have load-handling mechanisms. The problem is the difficulty to predict player density or deploy the players before the start of a game. Thus, the server cannot share its resources with surrounding region servers. This approach, presented a load balancing mechanism where region shapes are not predefined. Consider a system with a number of masters/servers with a large single region in the game world, the players can join and leave the game over time and at first only one server is involved. In this method [73], the game is divided dynamically depending on players' logical position and interaction pattern. Consequently, the zone shapes are not uniform. The bisection algorithm is used to divide a region into two sub-regions of roughly equal load while attempting to reduce the communication cost, such as the number of links crossing logical boundaries. Depending on the bisection methods, at the first time, the region is divided into one dimension to produce two sub-regions. Other partitions are made repeatedly in the new sub-regions if necessary [73]. However, this method is not efficient when the number of players is too large because we need extra servers to manage the load, then the cost of provide servers is high; furthermore, the maintenance of the servers is expensive.

3.9 Conclusion

This chapter discussed the state of the art with regard to the architecture that is used to design the MMOGs. It is introduced the key challenges of each architecture (client-server, pure P2P, and hybrid P2P). The advantages, disadvantages and restrictions of each method have been discussed and with the idea of proposing an approach that solves these limitations for MMOGs load balancing. Currently, all the researches relating to C/S architectures are focused on the methods to optimise and maximise of server resources to deal with more players; balancing the workload of servers in order to decrease latency; as of the result of server outages and network failure; as well as reducing the amount of over-provisioning of hardware, so the minimal hardware is apply in order to support player load. This is principally connected to the costs related to maintain of Client/Server infrastructure and is always cited as the essential influence for adopting P2P architectures for MMOGs.

However, P2P researches are focused on constructing solutions that can be used to distribute the load of MMOGs among all the players inside the game world in order to increase the scalability of the system. In addition to solve the problems that the decentralisation of the architecture creates. Decentralisation is an ideal method, a more realistic approach involves the use of hybrid architectures in which players dramatically contribute their resources to the overall system.

In both the P2P and Hybrid P2P architectures, research issues for example cheat-detection are just worth considering if it can be essentially demonstrated that a P2P messaging scheme can provide a good level of scalability, performance, responsiveness and consistency required to distribute game state updates. Therefore the gameplay experience is both enjoyable and maintainable. As a result, the main focus of research for MMOGs based on Hybrid P2P architectures must be the organisation of peers and the methods to distribute game state between them.

4

Hybrid Peer-to-Peer Architecture vs. Client/Server Architecture for Scalable MMOGs

4.1 Introduction

Network applications are considered some of the most significant sources of traffic in the network topologies. The traffic which is generated by network applications is used to make demands on the bandwidth and the underlying network technology, as well as to create load on the servers. In order to achieve the optimal performance of an application, we must ensure that the server and network infrastructure are designed to satisfy the application requirements. There are some criteria which must be taken into account when designing an application in the network topology in order to represent an accurate application. The transport protocol that is used in the network (e.g., TCP, UDP, RTP, etc.), and the number of concurrent connections, re-transmitter, failure and recovery, and so on. Each application in the network has its unique traffic pattern, and therefore creates its characteristic load on servers and the network. OPNET Modeler standard applications are remote login, Database, FTP, Peer-to-Peer file sharing, Email, video streaming, video conferencing, HTTP, Mobile user applications, print in application, and voice in application. In this chapter, we have designed and simulated MMOGs based on both hybrid P2P and client-server architectures by using OPNET Modeler standard applications. We have used TCP and UDP transport protocol in order to compare between them. The next section will explain the standard applications of OPNET Modeler that we used in our simulation.

4.2 Standard applications of OPNET Modeler

4.2.1 Remote Login Application Model

The Remote Login application model provides the connection between the user and a remote server in order to implement different operations on it by issuing commands from a local machine. Commands issued by a local system working together with the responses generated from the remote server to create traffic that moves through the network among the local and remote nodes [74]. In our simulation, the remote login application has been used to allow the players to register in the game server in order to create a player profile. All the players must be registered in the game server to enable them to start play the game.

4.2.2 Database Application Model

The database application is divided into two types, database entry and database query [74]. The database entry operation is responsible for writing data into the database. It consists of two messages, an entry message and a response message. An entry message is used to carry the data and a response message is used to carry the database acknowledgement of the operation. However, the database query operation is responsible for retrieving data from the database. It also consists of two messages a query message and response message. A query message carries the database request while a response message carries the data. In our simulation, using the database application model to save the players profile, players' queries, as well as the game state information, the query message size is always 512 bytes, however the size of the database response is dependent on the configuration of the transaction size attribute.

4.2.3 HTTP Application Model

The HTTP application model simulates web browsing activity. The client can periodically retrieve a page from a remote server. Each web page in OPNET Modeler contains text, image, graphic information, and sometimes video [74]. In our simulation, the player sends an HTTP demand for a web page. The game server receives the demand and sends the identical web page back to the player. If the page contains multiple information objects, thereafter the player node will request these objects from the server.

4.2.4 FTP Application Model

The FTP standard application model is considered one of the basic operations of the File Transfer Protocol between a client and a server. In OPNET Modeler, the regular FTP application has two basic commands for transferring a file: "put" and "get". The FTP put

operation is working to upload a file onto the FTP server, however the FTP get operation is used for downloading a file from the FTP server onto the client node [74]. The FTP application of OPNET Modeler is used to transfer one file at a time.

4.3 Design Scalable MMOGs based on Client/Server Architecture

The client/server architecture is considered one of the simple distributed systems that involve a separate client and server system, and network connection. Client/server architecture has been designed by using standard application of OPNET Modeler. We have designed a simple form of MMOGs client/server architecture by involving a server application that is accessed directly by several clients. The architecture consists of one centralised game server and a number of clients (players). All the players are connected together through a switch device. The switch device is directly connected to the game server through a client gateway router, an IP internet component, and a server gateway router. All the clients in the network are controlled by using the centralised game server. The game server is responsible for different functions such as players' registration in the game, game updating, storing all the information of players, controlling the communications between the players, managing the players' joining and leaving of the game. Figure 4.1 shows the design of client/server architecture.

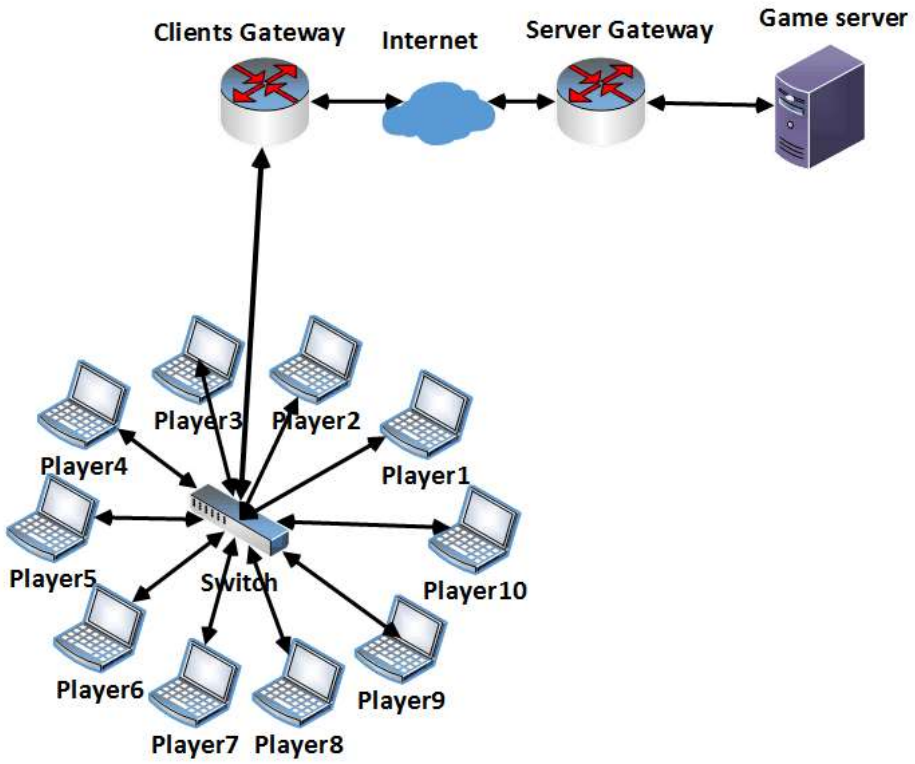


Figure 4. 1: Client/Server Architecture Design

4.4 Design Scalable MMOGs Based on Hybrid P2P Architecture

4.4.1 Design Requirements for MMOGs Hybrid P2P Architecture

There are some requirements which must be taken into account when designing a hybrid P2P architecture. These requirements are explained in the following section.

1- *Joining and leaving Requirement*: The player must connect to the game server to register in the game world.
2- *Scalability Requirement:* The system should be scalable with respect to the number of players that participate concurrently.
3- *Robustness Requirement:* The system should be robust when the number of players joining or leaving the game is too large.
4- *Responsiveness Requirement:* The system should always be efficiently responsive. This leads to obtain a good game performance.
5- *Distribution Requirement:* The system should be effective to distribute the players among the regions, distribute the control and operation cost between the participants evenly.
6- *Fairness Requirement:* The system should be providing the same quality of data, service and experience for all players in the virtual world.

4.4.2 MMOGs Hybrid P2P Architecture Design Concepts

In order to design MMOGs based on hybrid P2P system, there are three main concepts that should be explained before the design of our architecture. These concepts are described in the next section.

4.4.2.1 Game World Partitioning

In our research, the game world is divided into small areas called zones or regions. The main idea of dividing the game world of MMOGs into smaller portions is to make it more manageable. Each region is managed by both super-peer and clone-super-peer. The zone contains all the data associated with a zone. For instance all items located within that part of the game world, conditions of the environment, as well as all players and their particular information, like current position. The number of players peer zone depends entirely on the powerful resources of the super-peer and clone-super-peer such as (bandwidth, power CPU, and memory availability). These resources will be calculate by the centralised game server.

The major advantage of zoning is to achieve high scalability by distributing the players' load among several super-peers in the game world space. When a player migrates to another zone, the managing of the player will be the responsibility of the super-peer of the new region. The most important challenge in zoning method is overlapping, which happens when the player goes across zone borders. In this state, the player will be choosing a suitable super-peer to connect to it. Figure 4.2 illustrates the overlapping concept for players in the zone area.

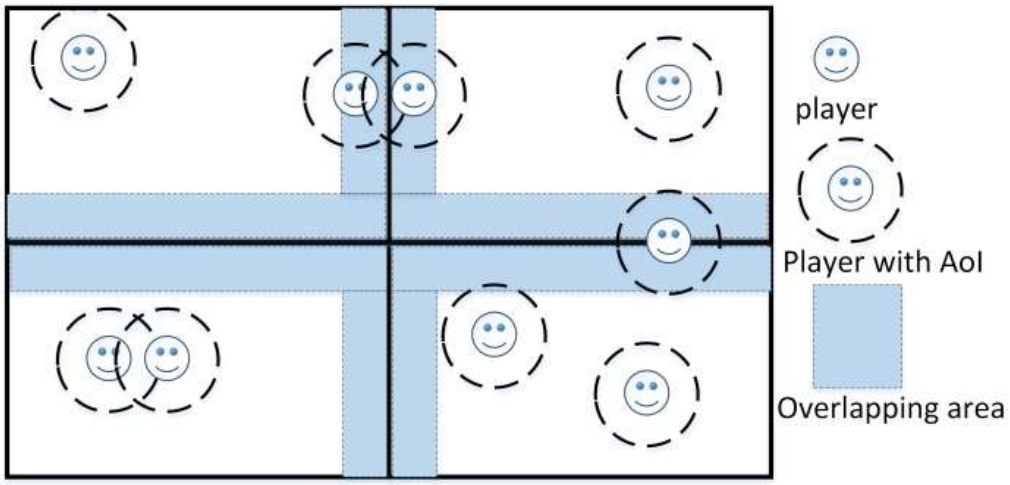


Figure 4. 2: The overlapping concept

4.4.2.2 The Super-peer Concept

In our research, we propose a new method for managing the zone using the clone-super-peer concept in addition to the normal super-peer. We have used the clone-super-peer concept for two reasons. The first reason is to avoid the single point of failure inside the region if used only the super-peer to manage and control the region. Therefore, the clone-super-peer will be the super-peer of the region when the current super-peer is crash or suddenly leave the game. The second reason is the clone-super-peer responsible for the migrating of the players from one region to another. This mechanism will help to reduce the load in the super-peer side. The super-peer used for updating the game state for all players in the zone, manages the message update dissemination and reduces the intercommunication, determines the Area of Interest (AoI) for each player in that region, calculates the resources of player such as (bandwidth, power CPU, and memory availability) to choose a better super-peer with powerful resources. This will help to cope with the overload of the zone by fragmenting the zone into two zones. Also, the super-peer then assigns a clone-super-peer of the region. However, clone-super-peer is used to deal with the migration of players from one region to another to avoid a region's super-peer being overloaded, which will make a region super-peer after the super-peer has left

the game or migrated to another region. It will also become a region super-peer when the region divides into two regions. The benefits of using the super-peer concept are the capability of becoming a local server for the zone by updating the game state and message dissemination for the players related to the zone. This will ensure the connection for the players, proposing the best organisation for the game world, minimizing the load of the centralised server by doing all the updates, calculating the load in the zone, and then dealing with the migration of players. There are some operations for the super-peer in the region of the game world described as follows:

- *Player joining*: the player attempts to join an appropriate super-peer by sending a request message to the region's super-peer. The super-peer accepts the request and sends a response message to inform the player that the connection is performed.
- *Publication*: the possibility of the player to publish some shared files by sending a request message to the region's super-peer. If the operations have been implemented accurately, the super-peer sends the response message to the player.
- *Search*: The player can search the contents of the region shared file if he/she wants to download it by sending a request message to the region's super-peer. The super-peer provides a list of available file and sends it by response message.
- *Player communication*: The super-peer is responsible for ensuring the connection between the players in the region without any fault. Because it is possible for the players to communicate with each other in the game world by sending a message to the region super-peer.
- *Super-peer communication*: The super-peer connects with other super-peers in the game world. This connection is used when the player migrate to another region.

4.4.2.3 The Zone State Concept

The zone state is totally dependent on the number of players that are connected to the super-peer in the region. There are three zone states in the game world: light zone state, overloaded zone state, and stable zone state. To find out the state of each zone, as mentioned before, the super-peer for the region calculates the resources and sets as threshold the number of players for the zone. If the number of players in the specified zone is equal or near a

threshold, the zone state is stable. However, if the number of players in the zone is higher than the threshold number, the zone's state is overloaded. Otherwise the state of the zone is light.

4.4.3 Design MMOGs Hybrid P2P Architecture

We have designed a novel architecture based on hybrid structured P2P overlay system to distribute the players among the regions, as well as to control the game state and provide more security to the system. The novelty of our architecture is the use of clone-super-peer in addition to the super-peer to manage and control the region of game world. This will help to reduce the load on the super-peer, as well as prevent any failure of the system when the super-peer is crash or suddenly leave the game. In addition to the mechanism of players' joining and leaving the game.

Our MMOGs architecture consists of a centralised game server, super-peer, clone-super-peer, and normal peers. The game world is divided into several manageable logical zones based on a geographical approach where each zone covers the players within this zone. The centralised server maintains the connections between super-peers and clone-super-peers, and prevents any failure in the game world. The zones are managed by both super-peer and clone-super-peer, which connect to the server to access the player's information. The responsibility of the super-peer are to preserve connections with the normal peers, updates the game state, manage the communication between the players inside the region, and manage the joining and leaving of the players. The clone-super-peer can be either a super-peer in controlling another region or a peer in the current region. Figure 4.3 shows the proposed MMOGs based on hybrid P2P architecture.

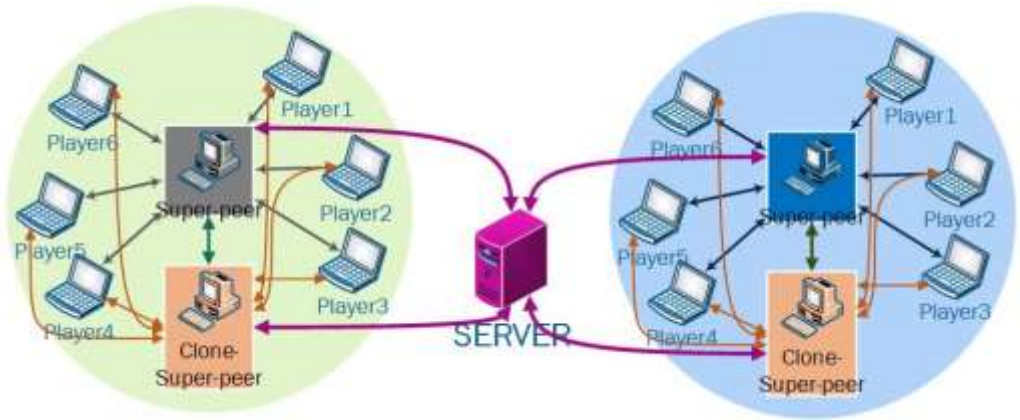


Figure 4. 3: The MMOGs Hybrid P2P Architecture

We are suggesting using hybrid architecture instead of centralised architecture and pure peer-to-peer for several reasons:

- Minimising the workload of the server, by sending message dissemination to the region super-peer.
- Saving the bandwidth of the server. This relates to the fact that players migration is done by the clone-super-peer without the need for the server, and the server sends the game state updates just for the region super-peers instead of all players in the game world.
- Decreasing the delay time on the server side, because most of the processes are done in the region without need for the server.

The most important aspects we must take into account when designing MMOGs based on hybrid P2P system are scalability and performance. The scalability aspect is mostly related to the maintenance and control that is required by the management protocol of P2P overlay. However, performance is related to the network metrics for instance bandwidth and the latency of the hybrid P2P architecture. There are two reasons for choosing hybrid P2P architecture instead of pure Peer-to-Peer architecture. These reasons are described as follows:

- *Security System*: The system security has become the key concern for developing games when the numbers of players increase rapidly. In the Peer-to-Peer system, preventing cheating players will be substantially more difficult compared with the centralised server architecture. It is important to know that the ordinary Peer doesn't have adequate resources to prevent cheating for a huge number of players in the game world. However, super-peer can be utilised to manage the game state in the region. In this case the security will apply mainly to the super-peer instead of all players in the game world.
- *Management System*: The management of a considerable number of players in a decentralised environment without a centralised system will be very difficult and probably not possible. For example, the management of consistency for MMOGs based on hybrid P2P system will be the responsibility of the game server in an AoIM. This consistency management principally is not much different from that existing in central server architecture.

4.4.4 MMOGs Hybrid P2P Architecture Mechanism

To understand the MMOGs based of hybrid P2P architecture, it is better to describe the mechanism that is relevant to the design.

4.4.4.1 The Mechanism of Joining the Players to the Game

The central server is responsible for registering the player when he/she joins the game as a first player or peer in the region, and assigning it a super-peer for the region. Also, the server assigns a unique key to this player. This key will be used in message update dissemination, game state update, and deployment of the players among the regions. Before the player joins the game, they must decide which zone they want to play in. This process gives the player more flexibility to choose other participants to interact with them. After the players register in the game, the super-peer of the zone checks the hardware resources of the player such as (CPU power, memory available, and bandwidth) to see which peer has powerful resources to be a super-peer and clone-super-peer for the zone. This process will be done when the region's super-peer or clone-super-peer has left the game. The region's super-peer stores all the information of players belonging to your zone in a database in the game server. The clone-super-peer has three main benefits such as becoming the super-peer of the region after the super-peer has left the game or migrate to another region, deal with the migration of players, and becoming a super-peer when the zone divides into two zones due to overload. Figure 4.4 shows the mechanism of joining and leaving the game.

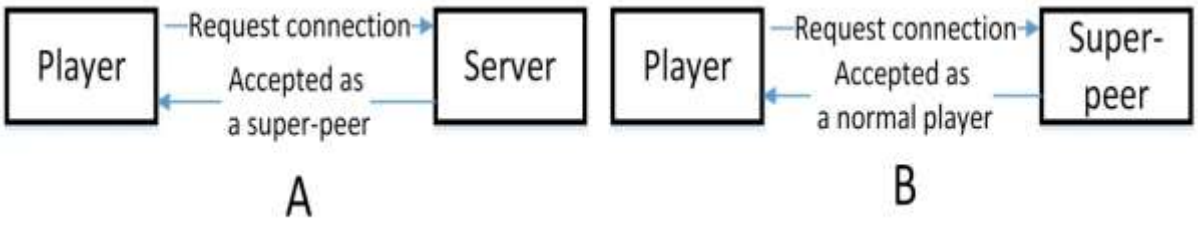


Figure 4. 4: (A) Player joins as a first player, (B) Joining normal player.

4.4.4.2 The Mechanism of the Migration of Players

The player in the game world has a freedom to move from his/her region to another region. To migrate, the player finds an appropriate zone and informs the clone-super-peer about it by sending the zone keyID. The clone-super-peer checks the state of the new zone. If the state of the zone is light load, the clone-super-peer informs the peer to migrate, but if the state

of the new zone is overload, the clone-super-peer migrates the peer to the new region and divides the new region into two regions. Figure 4.5 illustrates the player migration.

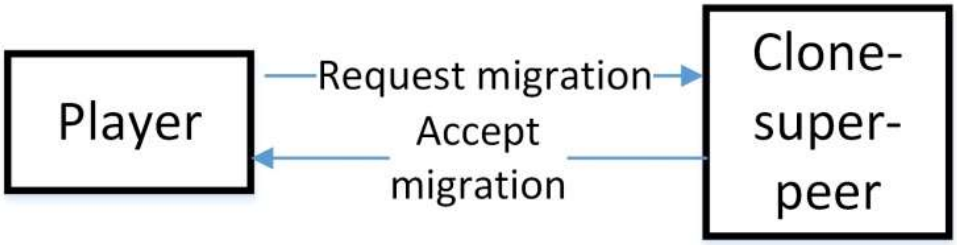


Figure 4. 5: Player Migration

4.4.4.3 The Mechanism of the Players Leaving the Game

When the normal peer wants to leave the game, they must inform the region's super-peer to save his/her profile and decrement the total number of players in the region. However, when the clone-super-peer wants to leave the game, he must also inform the region's super-peer to set an alternative clone-super-peer for the region, as well as informing the game server of that. Whereas, a super-peer leaving is done by informing both the server and the clone-super-peer and switching the clone-super-peer to become the region super-peer or to organise the election of the new super-peer for that region. Also, the new super-peer assigns a clone-super-peer to that region if none is currently assigned. However, if the region super-peer suddenly leaves the game, the clone-super-peer automatically becomes the super-peer for the zone. Figure 4.6 illustrates the player leaving the game.

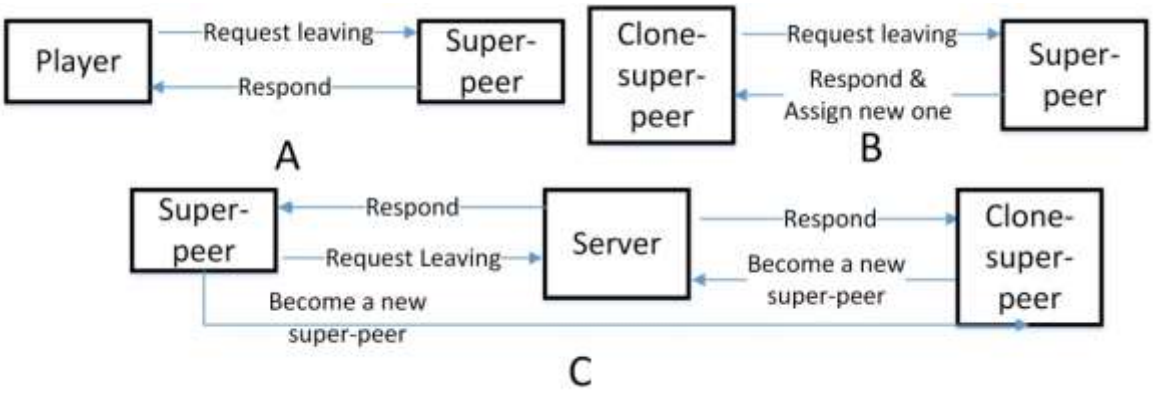


Figure 4. 6: player leaving the game, (A) normal player leaving, (B) clone-super-peer leaving (C) super-peer leaving.

4.4.4.4 The Mechanism of Players' Deployment

When all of the super-peers and clone-super-peers are assigned, they must calculate the resources available to them, such as CPU power, memory available, and bandwidth. This calculation is used to determine the number of peers that will be connected to it. The number of peers per region must be known by the server and the local super-peer, as well as the clone-super-peer. In addition, the local super-peer and clone-super-peer have a list of super-peers related to the game world that is used when a player migrate to another zone. The current super-peer can evaluate the resources of each player connected to it to find the peer which has the most powerful resources to be the new super-peer node. This process is important to ensure the scalability of the system. Deployment of the peers will be the responsibility of the centralised server by sending it to the region super-peer. When the player joins the game, he/she will choose the zone where they want to play. The server is responsible for controlling all the super-peers for the game world to ensure both the organisation and the performance of the game world. As well as this architecture will achieve less latency without the need for high bandwidth.

4.5 System Evaluation

To evaluate the scalability of both client/server and hybrid P2P architecture, we have simulated the two architectures using the OPNET Modeler 18.0 simulation environment [42]. The next section explains the network simulation.

4.5.1 Network Simulation

It is an important phase of our research to perform the implementation and evaluation of scalable MMOGs based on hybrid P2P architecture. The main target of our project simulation is to evaluate our proposed framework and confirm its effectiveness in the simulation environment. Simulation becomes the only viable alternative to evaluate, understand, and choose between design technique, and network protocols for very large distributed systems involving thousands of concurrent nodes. Simulation will help us to get important information on feasibility and practicality which are crucial to the implementation of the system prior to investing considerable time and money on real systems. The proper design of the network architecture in MMOGs is critical, due to the huge number of concurrent players, this leads to very complex network architectures. When the networks complexity increases, the design and analysis of network become more challenging. The complexity of MMOGs network architecture has made the use of simulation approaches necessary for the

analysis and evaluation of the performance of the game network. There are different aspects in MMOGs game network which can be analysed and evaluated by using the network simulation approaches. These aspects are described as follows:

- **The architecture for MMOG game server**: Network simulation methods can be used to analyse the game server architecture, and check the system performance when you use a hybrid peer-to-peer architecture. As well as evaluating the latency value in the game network, and its impact on the performance of the game.
- **Client resources**: Network simulation methods can be used to find out the resources of the server such as the power of the CPU, the bandwidth, and memory availability, and check which resource may have an effect on the scalability, consistency, latency, and fault-tolerance.
- **Protocol of the game network**: Can be analysed and verified for the performance of all the network protocol used in the game by using network simulation methods.
- **AoI Management performance**: Several AoI techniques are used by MMOGs to minimise the volume of game network traffic. By using a network simulation approach, the performance of various AoI techniques can be measured and developed. So, it is the best method to determine optimal AoI techniques for the MMOGs.
- **Game world partitioning**: Typically, MMOGs divide the virtual environment into regions to distribute the players' load among super-peers. Network simulation approaches can determine an optimum solution to divide the virtual game world with minimum latency values.
- **Dynamic load balancing**: Network simulation methods can be used to analyse the load of the network and check the overloaded region, as well as evaluate the load balancing technique that used to deal with the overloaded region.

4.5.2 Simulation Parameters

- **Number of Nodes**
  - In our simulation, we have used 125, 500 and 1000 nodes for each scenario.
- **Types of Nodes**

- We have used Ethernet for design wired network. However, used WLAN for design wireless network. We have TDMA for design AoIM network and Load balancing network.

- **Number of Regions**
  - The number of regions for 125 nodes are 2 regions. While, we have 8 regions for 500 nodes. However, the number of regions for 1000 nodes are 16 regions.
- **Simulation Time**
  - 300 second for 125 nodes scenario
  - 600 second for 500 nodes scenario
  - 1200 second for 1000 nodes scenario
- **Transport protocols**
  - Transmission Control Protocol (TCP)
  - User Datagram Protocol (UDP)

4.5.3 Simulation of MMOGs Client/Server Architecture

Using OPNET Modeler 18.0 to simulate the architecture that was designed in section 4.3. The client/server architecture consists of the following devices: game server, two Cisco C400 Router, IP Cloud, Ethernet Switch, and Ethernet workstations as shown in figure 4.7. The connection mechanism between these devices is described as follows: The game server connects with the IP Cloud through Cisco C400 Router (server gateway). The IP cloud connects with the Ethernet Workstations (Peers) through both Cisco C400 Router (Client gateway) and Ethernet Switch. Using Ethernet 100BaseT link to connect between server and server gateway, also between the client gateway and Ethernet switch. Using PPP DS1 link to connect between Router and IP Cloud, also between IP Cloud and Client gateway. Use Ethernet 10BaseT link to connect between the Ethernet Workstations and Ethernet Switch.

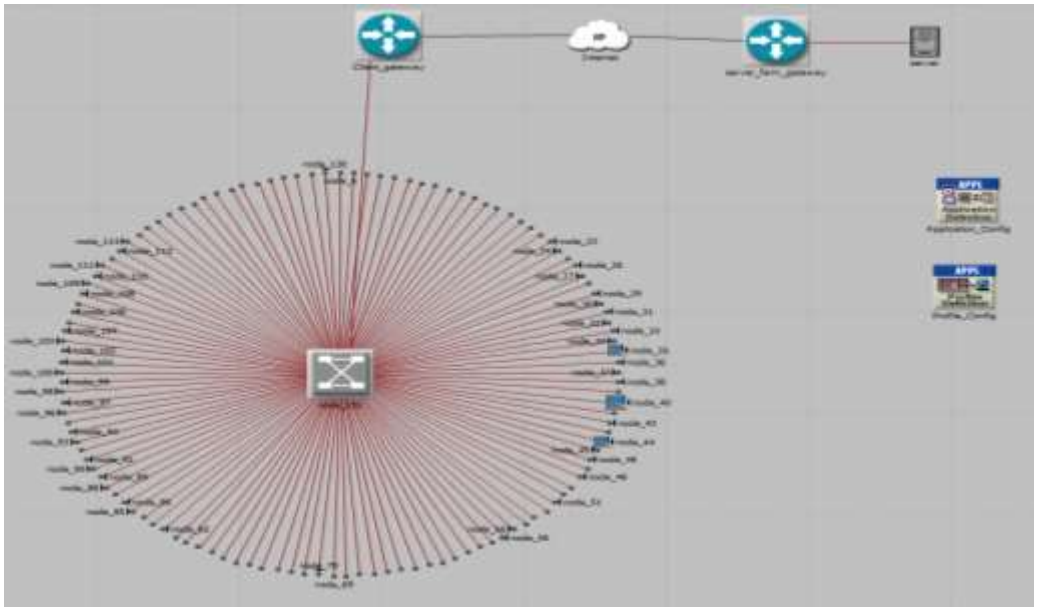


Figure 4. 7: MMOGs Client/Server Architecture Scenario

In Client/Server architecture, all the applications that are shown in table 4.1 are managed and controlled by the server. Table 4.1 also shows the results parameters and units that were used in the simulations. Using default TCP transport protocol to simulate all the scenarios.

Table 4. 1: The Applications Used in Simulation

Applications Name	Start Time Offset (second)	Duration (second)	Parameters	Unite
Remote Login	constant (1)	constant (300)	Delay Traffic Received	Second Bytes/sec
Database Access	constant (1)	constant (300)	Delay Traffic Received	Second Bytes/sec
HTTP	constant (1)	constant (300)	Delay Traffic Received	Second Bytes/sec
FTP	constant (1)	constant (300)	Delay Traffic Received	Second Bytes/sec

The simulation consists of eight scenarios. These scenarios are explained in the following sections:

- Client/server with 125 peers without background traffic.
- Client/server with 250 peers without background traffic.
- Client/server with 500 peers without background traffic.
- Client/server with 1000 peers without background traffic.

4.5.4 Simulation of MMOGs Hybrid P2P Architecture

The simulation of hybrid P2P MMOGs infrastructure is used to ensure the architecture that is used in MMOGs provides a high level of scalability and consistency, with a low level of latency, and fault-tolerance. In order to demonstrate our MMOGs hybrid P2P architecture, we will compare the results with the MMOGs Client/Server architecture. MMOGs based on hybrid P2P system consists of the same devices in MMOGs Client/Server system scenario but with a little bit of change in the region area by adding both super-peer and clone-super-peer for each region. Each region has 60 that peers are connected to both super-peer and clone-super-peer using the super-peer gateway router as shown in figures 4.8, 4.9 and 4.10. In MMOGs hybrid P2P system, only the remote login is managed and controlled by the game server, the other applications are managed and control by the using super-peer and clone-super-peer of the region.

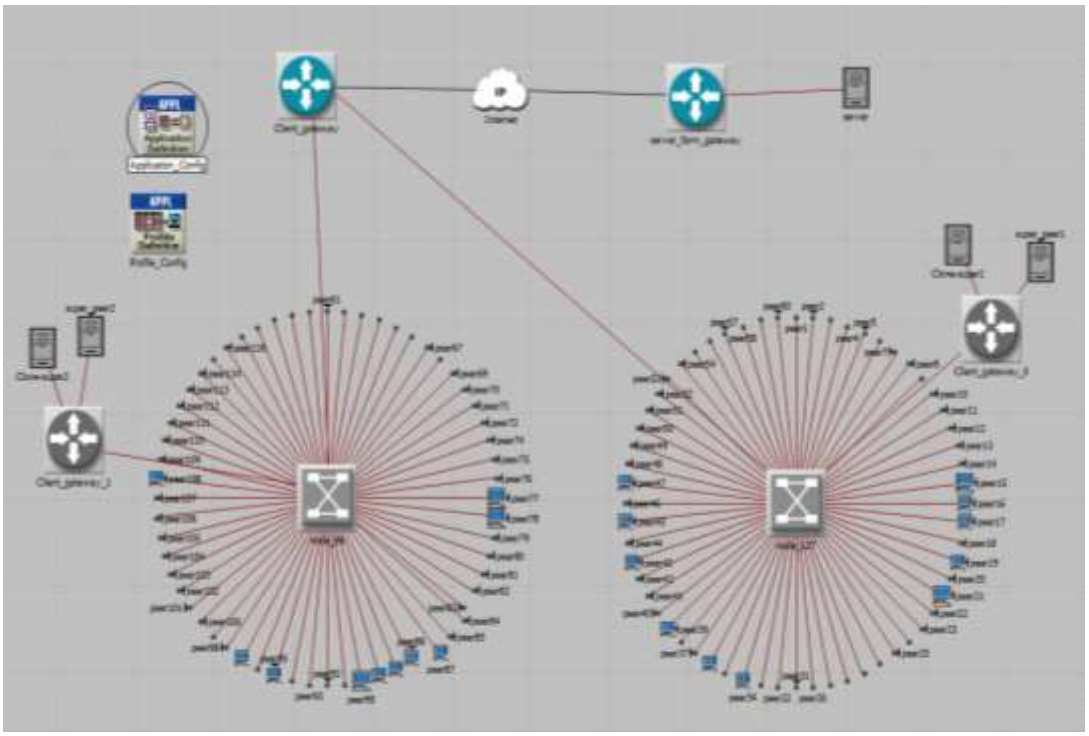


Figure 4. 8: MMOGs Hybrid P2P Architecture scenario with 125 peers

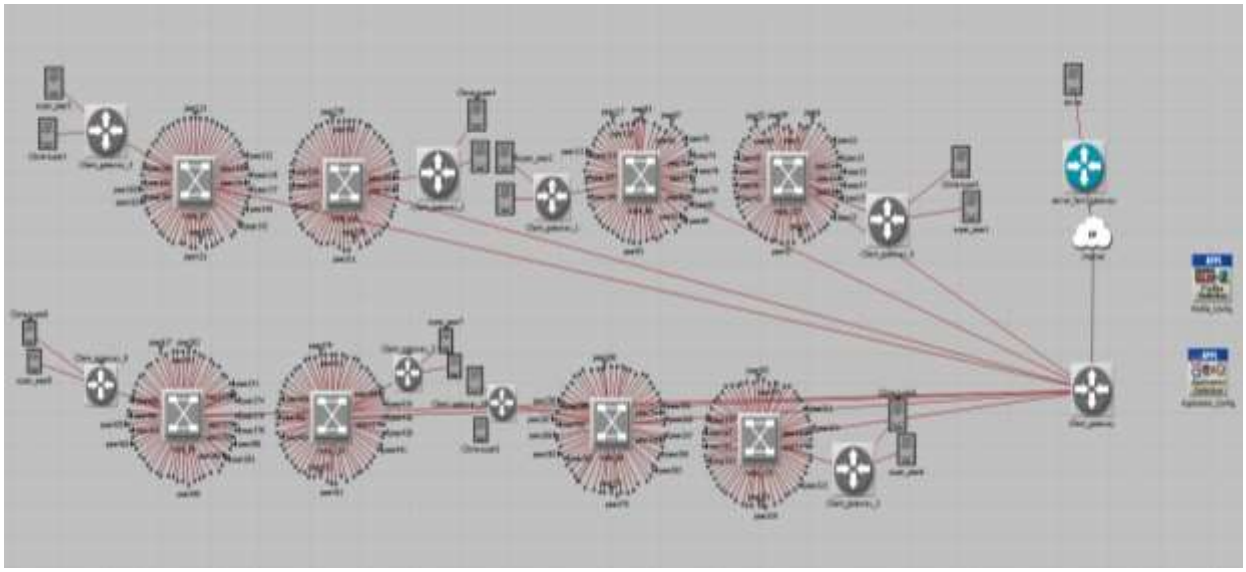


Figure 4. 9: MMOGs Hybrid P2P Architecture scenario with 500 peers

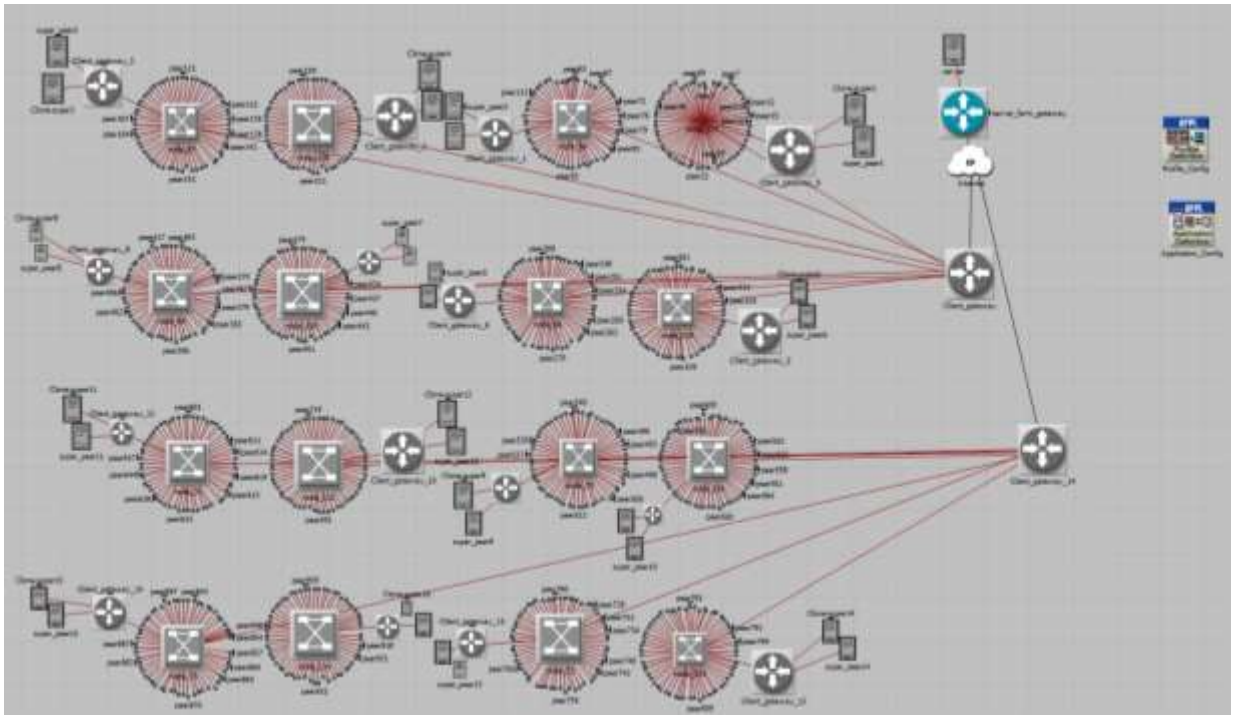


Figure 4. 10: MMOGs Hybrid P2P Architecture scenario with 1000 peers

4.5.5 Simulation Results and Analysis

All the scenarios for both architectures are simulated using OPNET 18.0 Modeler. The computer used for the simulation is Intel core(TM) i7-3770 CPU@3.40GHZ, 32.0 GB RAM, and 64-bit operating System. All the parameter results that were used in the simulation are explained in the following section.

4.5.5.1 Overall Delay

This parameter is defined as the overall end-to-end delay for all packets received by the server. End- to-end delay for the application that were used during the simulation is measured the time from source to destination. Figures 4.11, 4.12, 4.13, and 4.14 show the overall delay for the hybrid P2P architecture compared with the client/server overall delay of the architecture with 125, 250, 500, and 1,000 peers respectively without adding any background traffic to the connection link. In the figure below, we have used four OPNET custom applications to represent the game application. These applications are Remote Login, Database, HTTP and FTP respectively. We have given 5 minutes for each application to simulate. The figures below illustrate that the delay for the client/server system is greater than the delay of the hybrid P2P system for all application used in the simulation except the remote login application because this application is the same for both architectures. The figure start by the Remote Login application and continue for 5 minutes. All the players must login in the game server at the first time in both hybrid P2P and Client-Server architectures. This is the reason that makes the delay nearly equal for the remote login application for all four scenarios. After the Remote Login application finished, the Database application start to work for 5 minutes also. The database query delay for client/server system is four times greater than the delay of database query in hybrid P2P system. Due to all players in client/server architecture sending their queries of database directly to the server, however, the players in hybrid P2P architecture send queries of databases to the region's super-peer and clone-super-peer, and this helps to decrease the overall delay for hybrid P2P system. After the Database application have finished, the HTTP start to work for 5 minutes. The delay produced by HTTP for the hybrid P2P architecture is less than the HTTP delay for client-server architecture. The last application is FTP and work for 5 minutes. The delay produced by FTP application for client-server system is significantly large compared to the FTP delay of hybrid P2P architecture. Because all the files are requested from the game server in the client-server system. However, the files are requested by using the region’s super-peer in the hybrid P2P system.

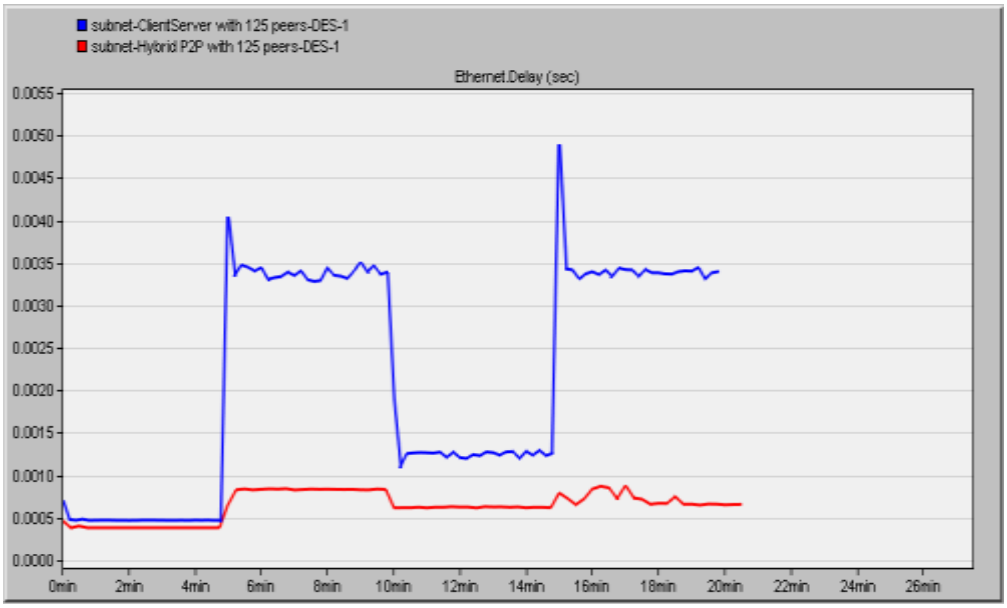


Figure 4. 11: The Overall Delay for both MMOGs Hybrid P2P and Client/Server with 125 peers

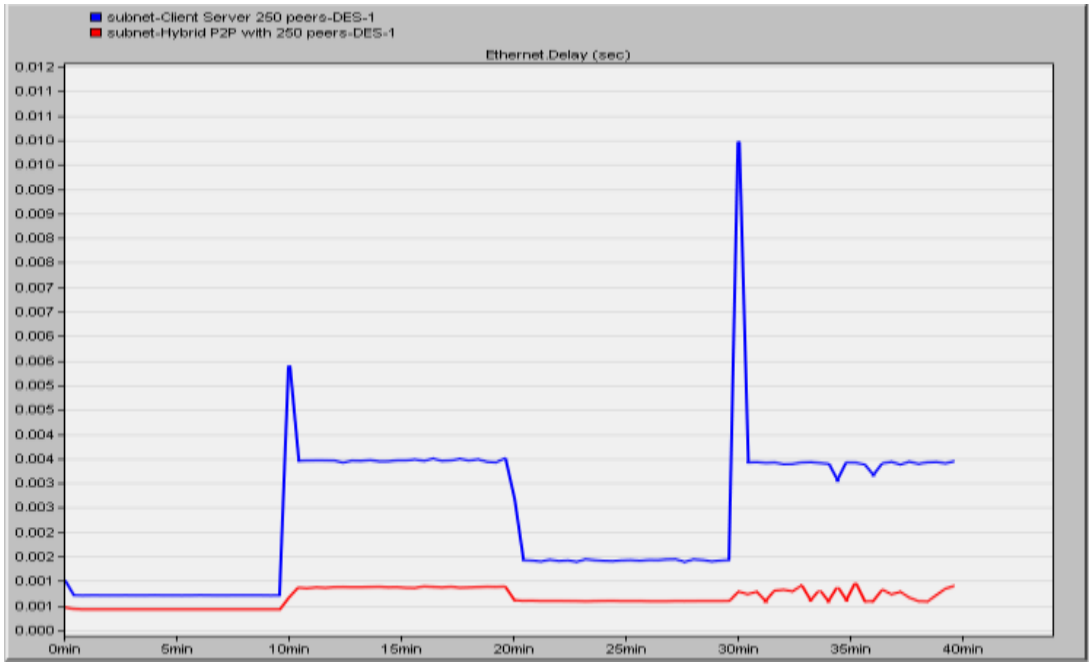


Figure 4. 12: The Overall Delay for both MMOGs Hybrid P2P and Client/Server with 250 peers

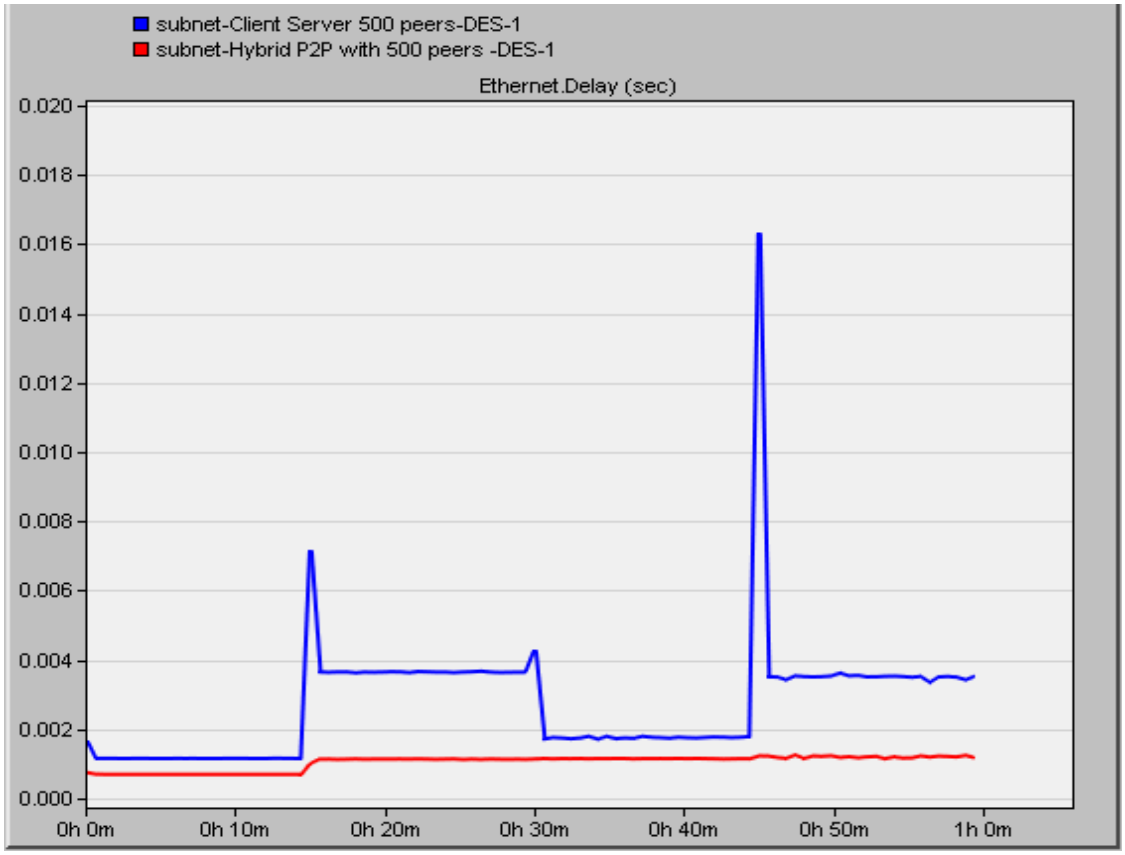


Figure 4. 13: The Overall Delay for both MMOGs Hybrid P2P and Client/Server with 500 peers

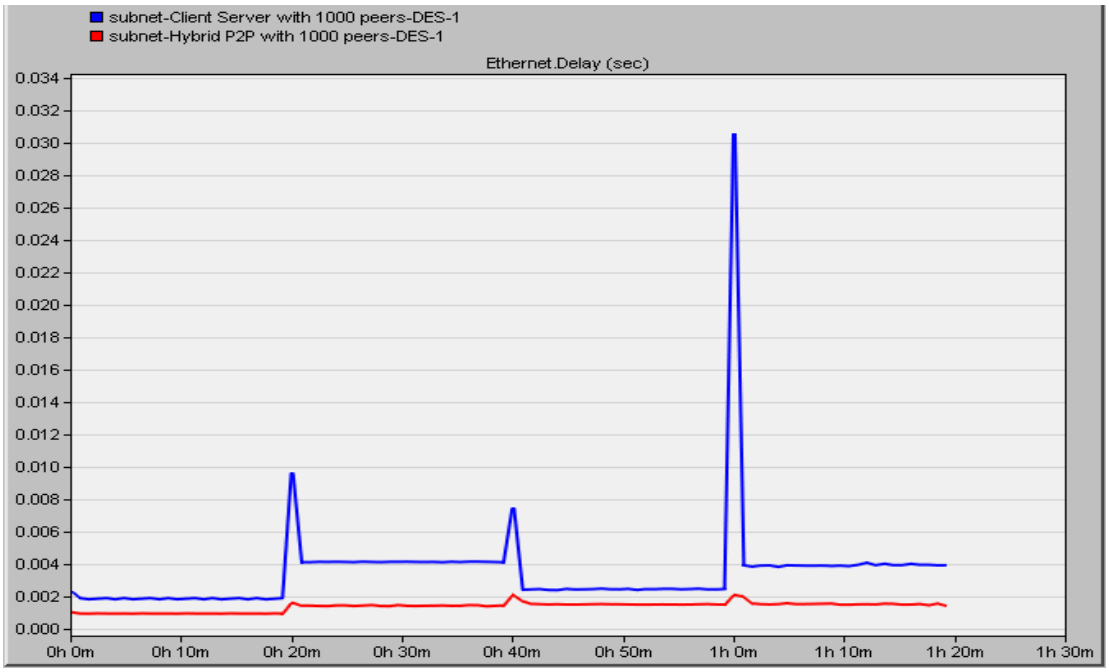


Figure 4. 14: The Overall Delay for both MMOGs Hybrid P2P and Client/Server with 1000 peers

4.5.5.2 Traffic Received for Remote Login

The Remote Login application allows the user to connect with a remote server and implements different operations on it by issuing commands from a local machine. However, traffic received is the average number of bytes per second forwarded to all remote login applications by the transport layers in the network. Figures 4.15, 4.16, 4.17 and 4.18 illustrate the traffic received for the remote login application with 125, 250, 500, and 1000 peers respectively for both client/server and hybrid P2P architecture. The figures show that the traffic received for the remote login application is nearly equal, due to the remote login being managed by the server for both hybrid P2P and client-server architecture. However, there is a very large variation between the traffic Received for Remote Login for scenario with 500 peers and the scenario with 1,000 peers for both architectures, up to five times. These variations in the traffic received are caused by the standard applications because there is no method to control and manage the players during the simulation compared to the custom applications.

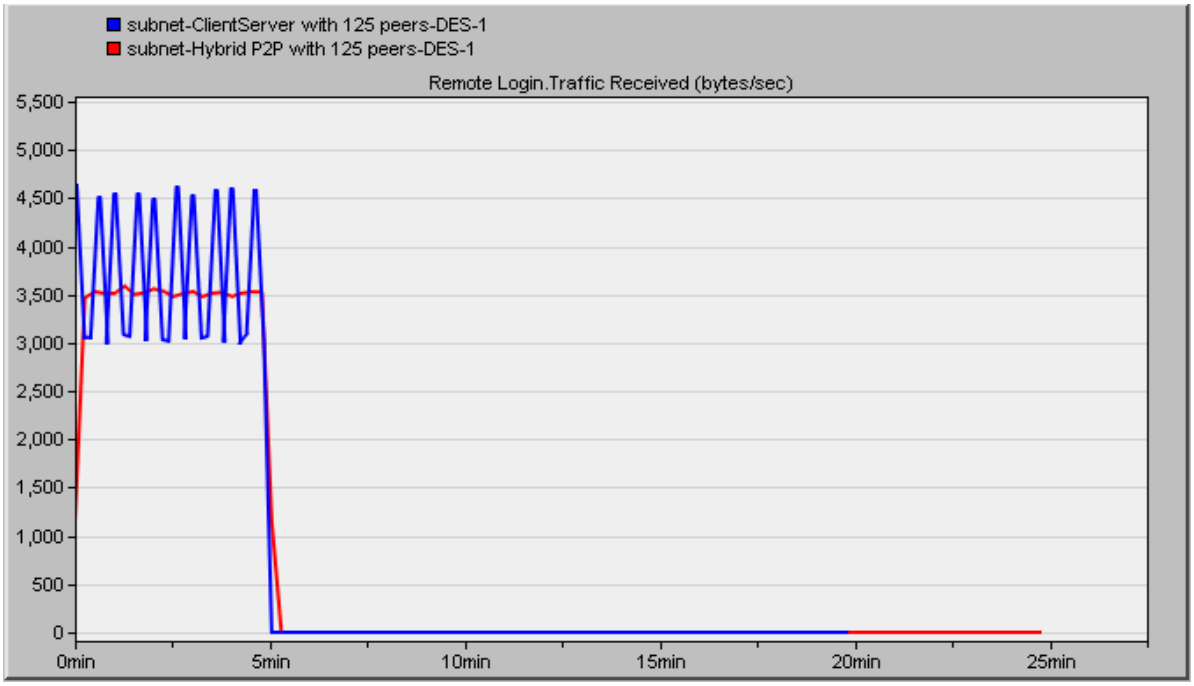


Figure 4. 15: The Traffic Received of Remote Login for both MMOGs Hybrid P2P and Client/Server with 125 peers

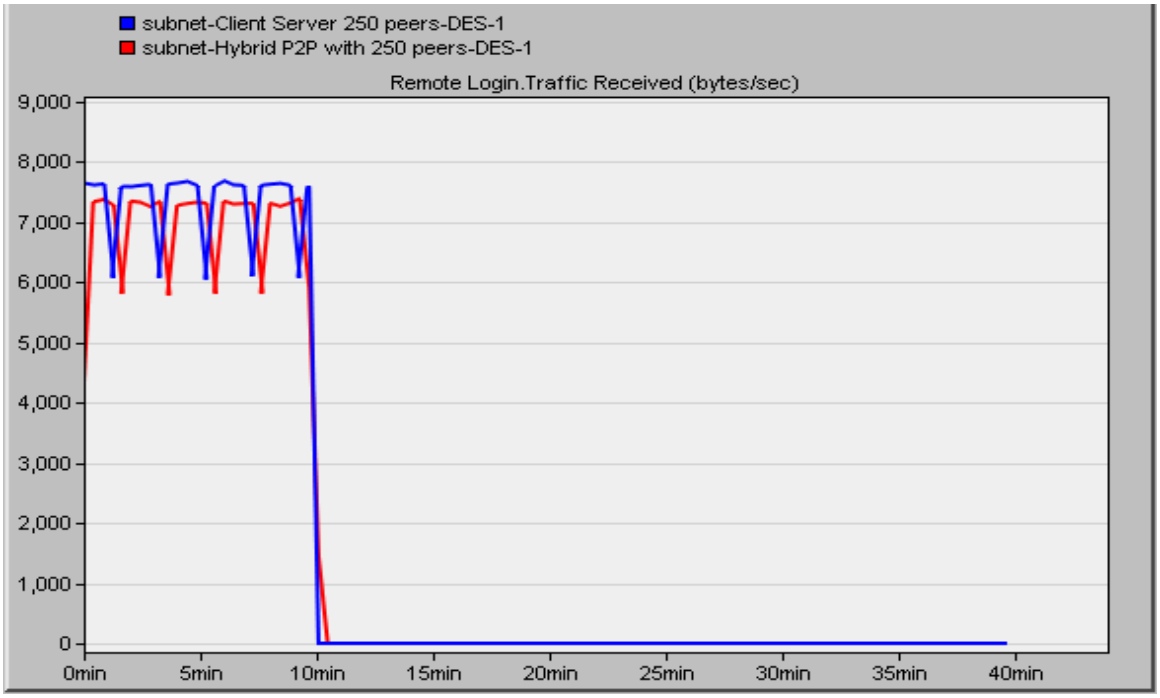


Figure 4. 16: The Traffic Received of Remote Login for both MMOGs Hybrid P2P and Client/Server with 250 peers

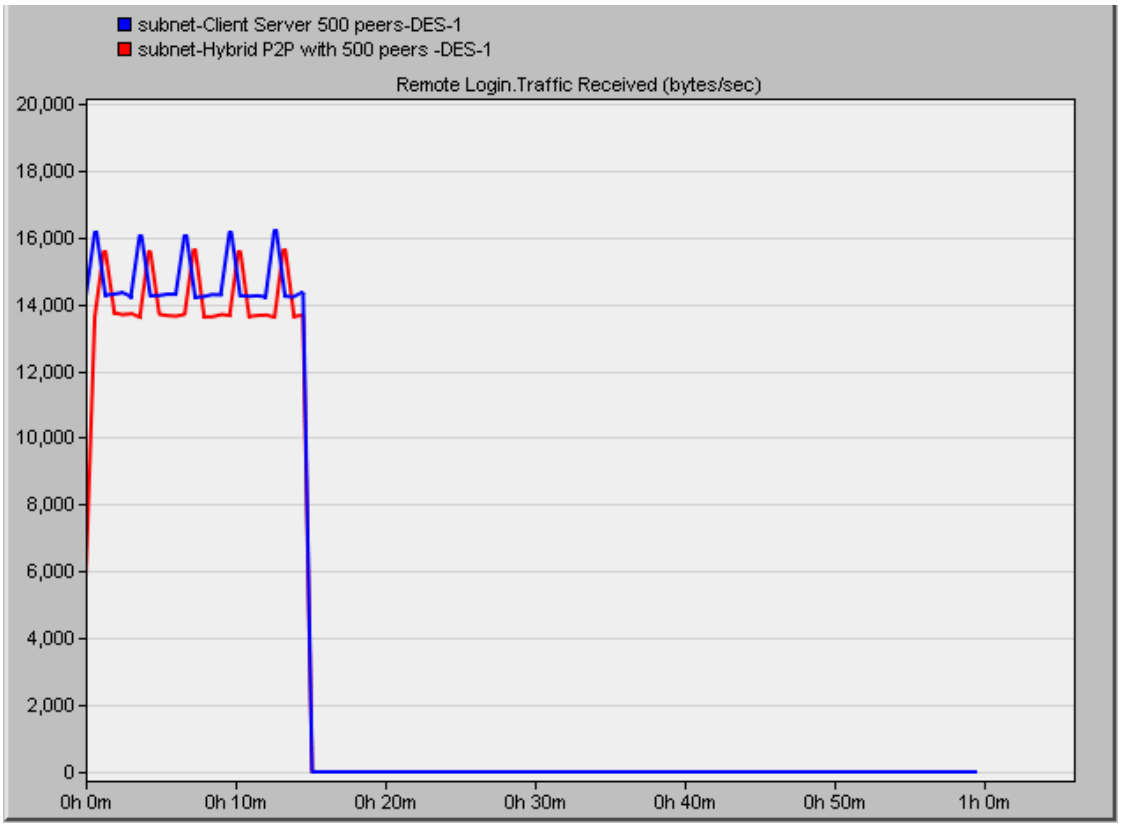


Figure 4. 17: The Traffic Received of Remote Login for both MMOGs Hybrid P2P and Client/Server with 500 peers

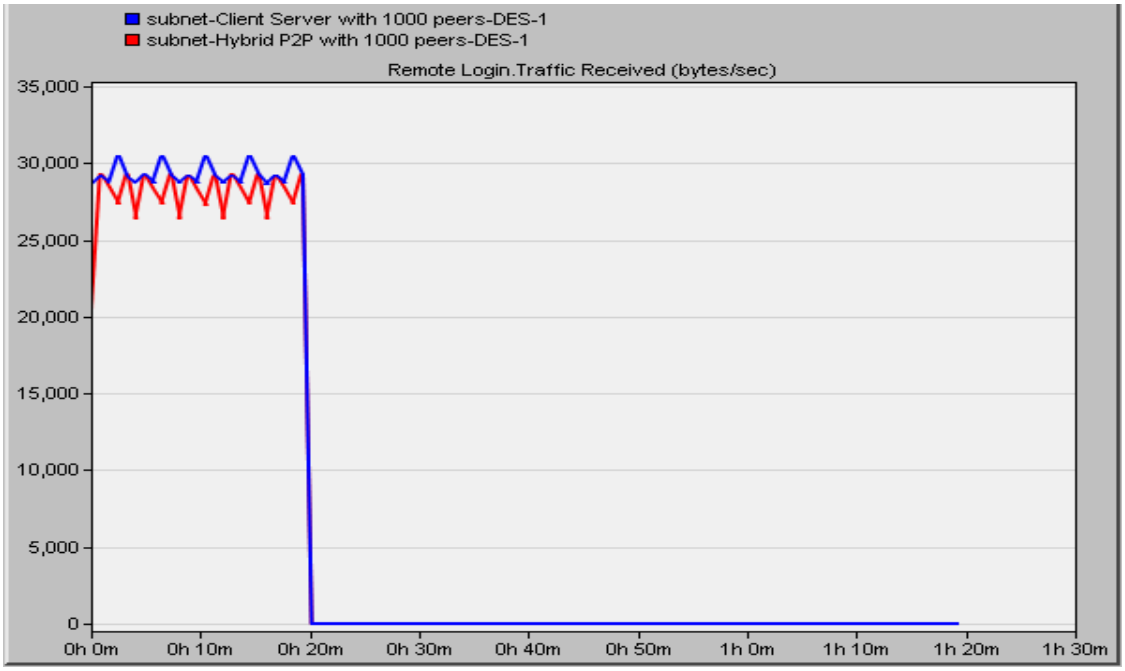


Figure 4. 18: The Traffic Received of Remote Login for both MMOGs Hybrid P2P and Client/Server with 1000 peers

4.5.5.3 Traffic Received for Database Query

The database query operates on retrieving data from the database. It consists of a query message that holds the database request and a response message that holds the data. The traffic received for database query defined as the average number of bytes per second forwarded to all database query applications by the transport layers in the network. Figures 4.19, 4.20, 4.21 and 4.22 display the traffic received for database query with 125, 250, 500, and 1000 peers respectively for both client/server and hybrid P2P architecture. The figures explain that the traffic received of database query for client/server system is much greater than the traffic received of database query for hybrid P2P system up to nearly five times with 500 peers. However, traffic received for database query up to highest level is more than ten times in client/server architecture with 1,000 peers compared with hybrid P2P with same number of peers. This shows that when the number of players increases, the traffic received of database query will increase while it remains stable in hybrid P2P architecture. This is because the database query in hybrid P2P system is managed and controlled by the region super-peer and clone-super-peer.

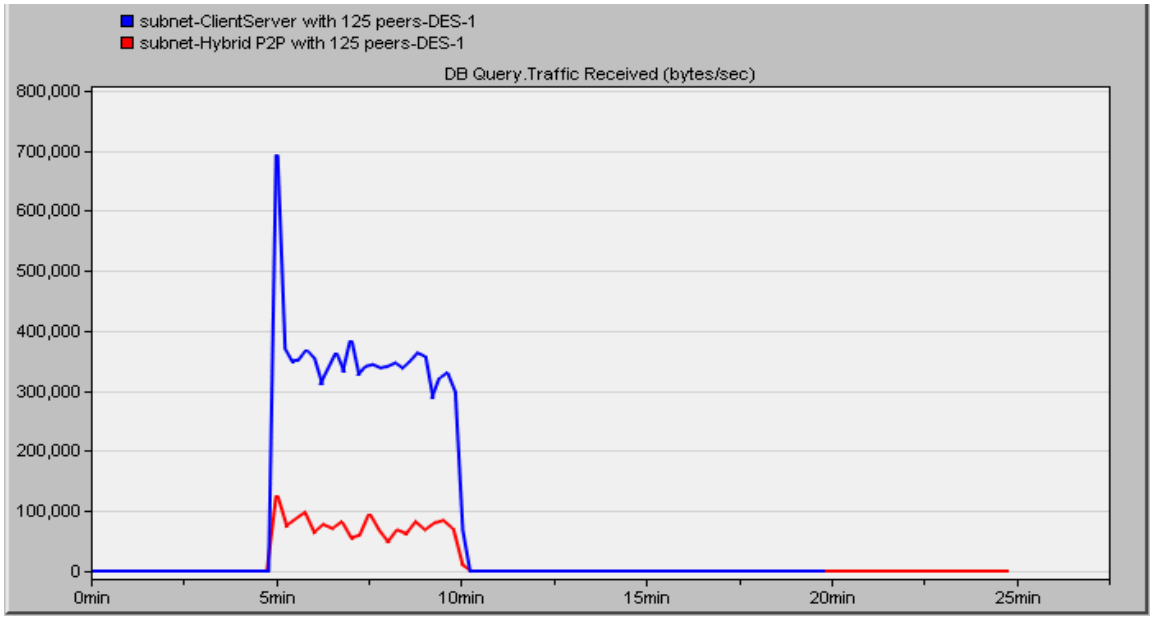


Figure 4. 19: The Traffic Received of Database Query for both MMOGs Hybrid P2P and Client/Server with 125 peers

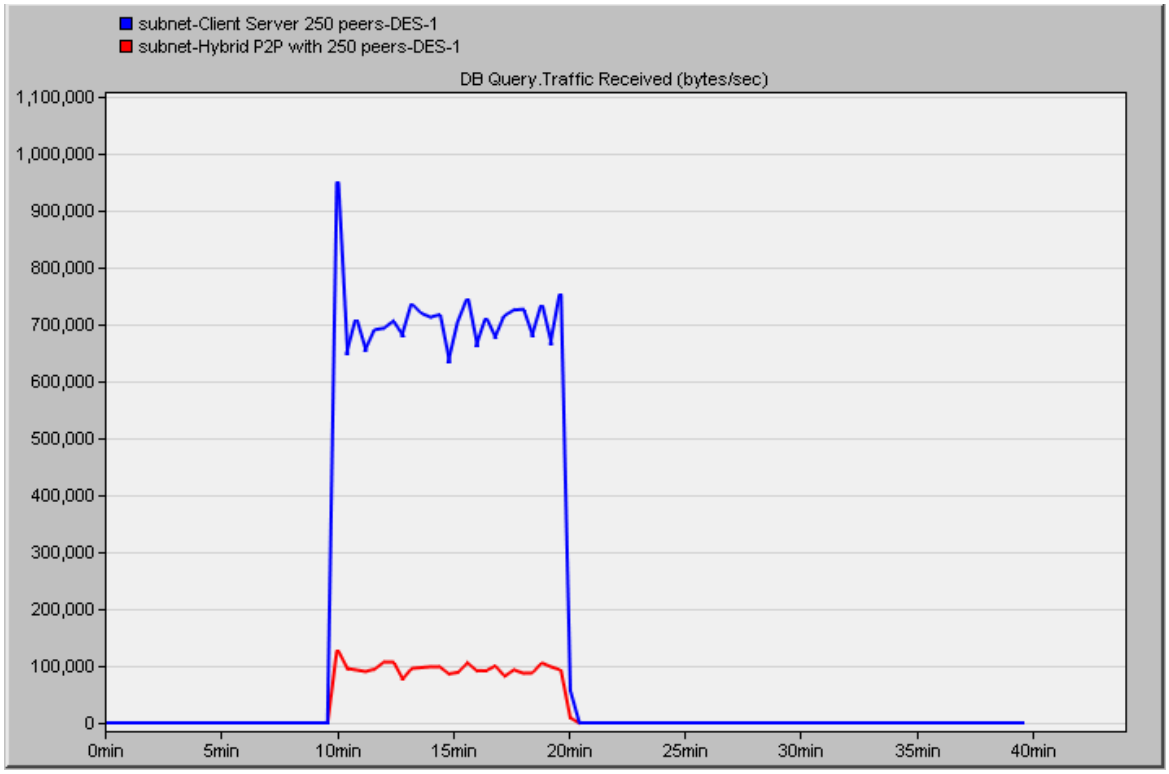


Figure 4. 20: The Traffic Received of Database Query for both MMOGs Hybrid P2P and Client/Server with 250 peers

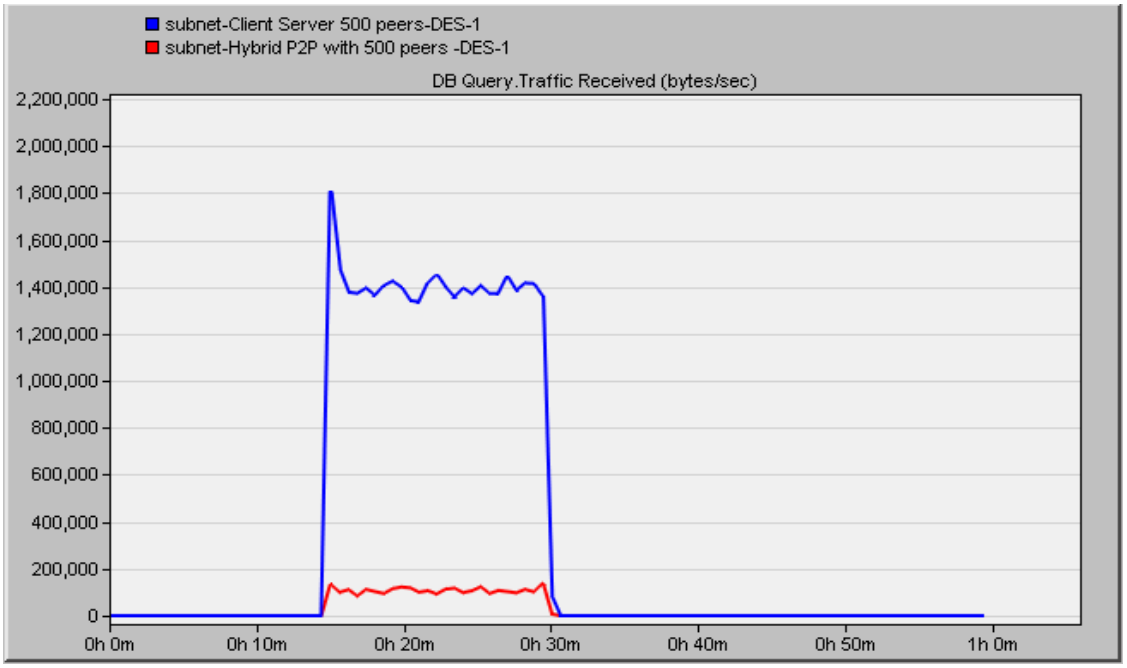


Figure 4. 21: The Traffic Received of Database Query for both MMOGs Hybrid P2P and Client/Server with 500 peers

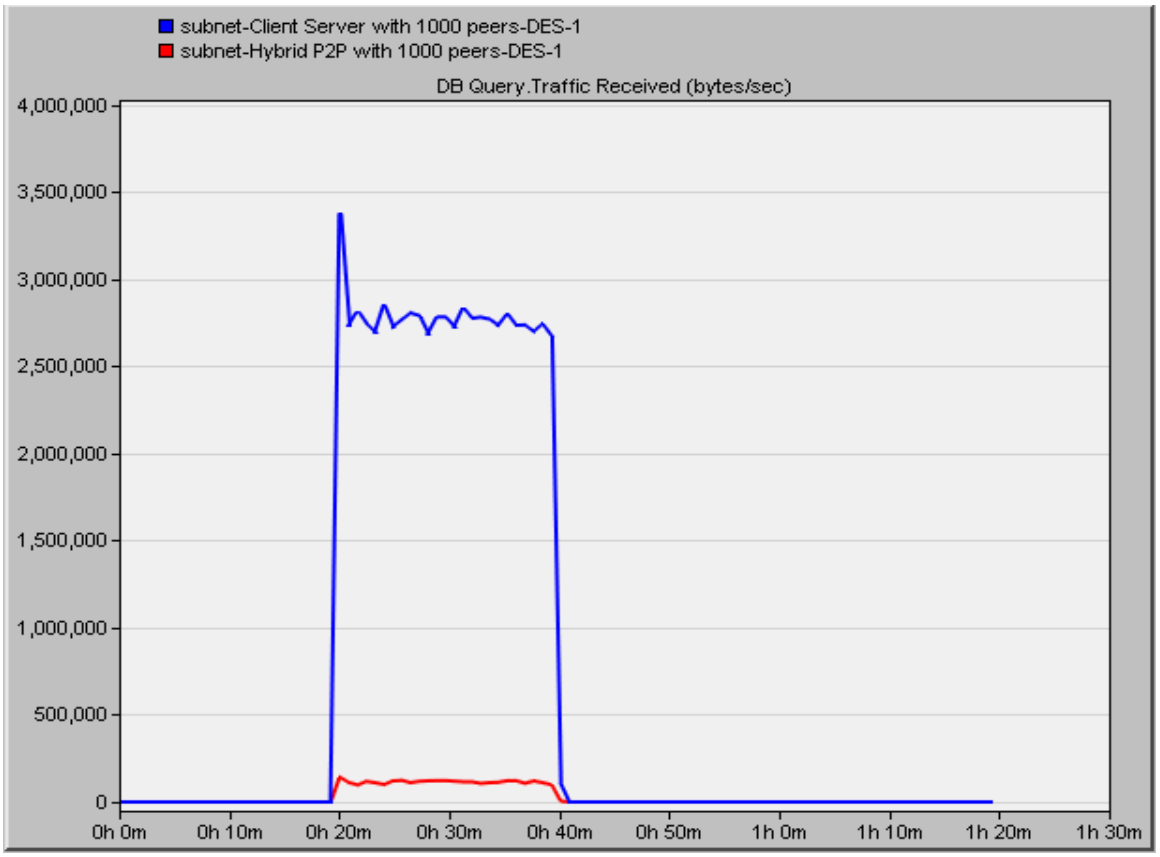


Figure 4. 22: The Traffic Received of Database Query for both MMOGs Hybrid P2P and Client/Server with 1000 peers

4.5.5.4 Traffic Received for HTTP

The HyperText Transfer Protocol (HTTP) is the Internet browsing activity where a client node periodically contacts web servers to get web pages. HTTP traffic received is the average number of bytes per second forwarded to the HTTP application by the transport layers in the network. Figures 4.23, 4.24, 4.25 and 4.26 illustrate clearly the difference between the HTTP traffic received for client/server and hybrid P2P system with 125, 250, 500, and 1,000 peers respectively. The figures below show clearly the increase of HTTP traffic received to reach the highest peak with 340.000 bytes/second in the scenario four with 1000 peers, while it remains stable in hybrid P2P architecture for all scenarios by less than 20.000 bytes/second.

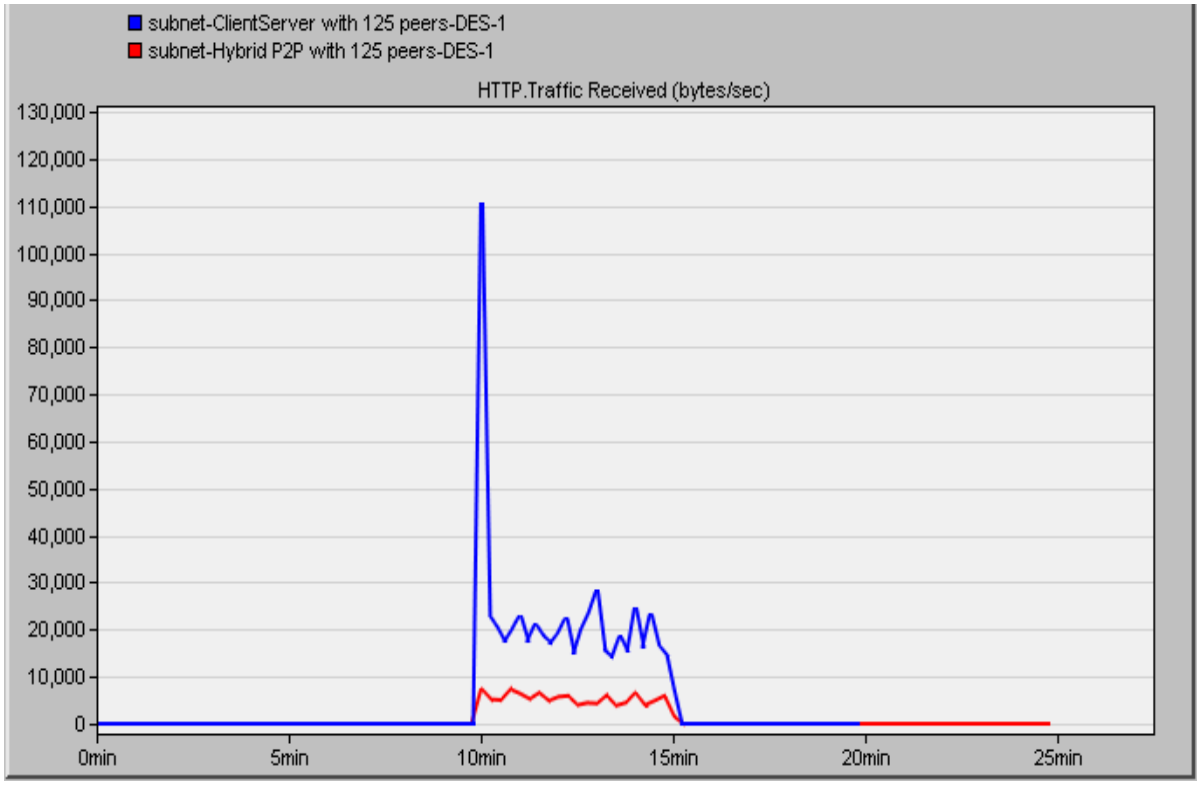


Figure 4. 23: The Traffic Received of HTTP for both MMOGs Hybrid P2P and Client/Server with 125 peers

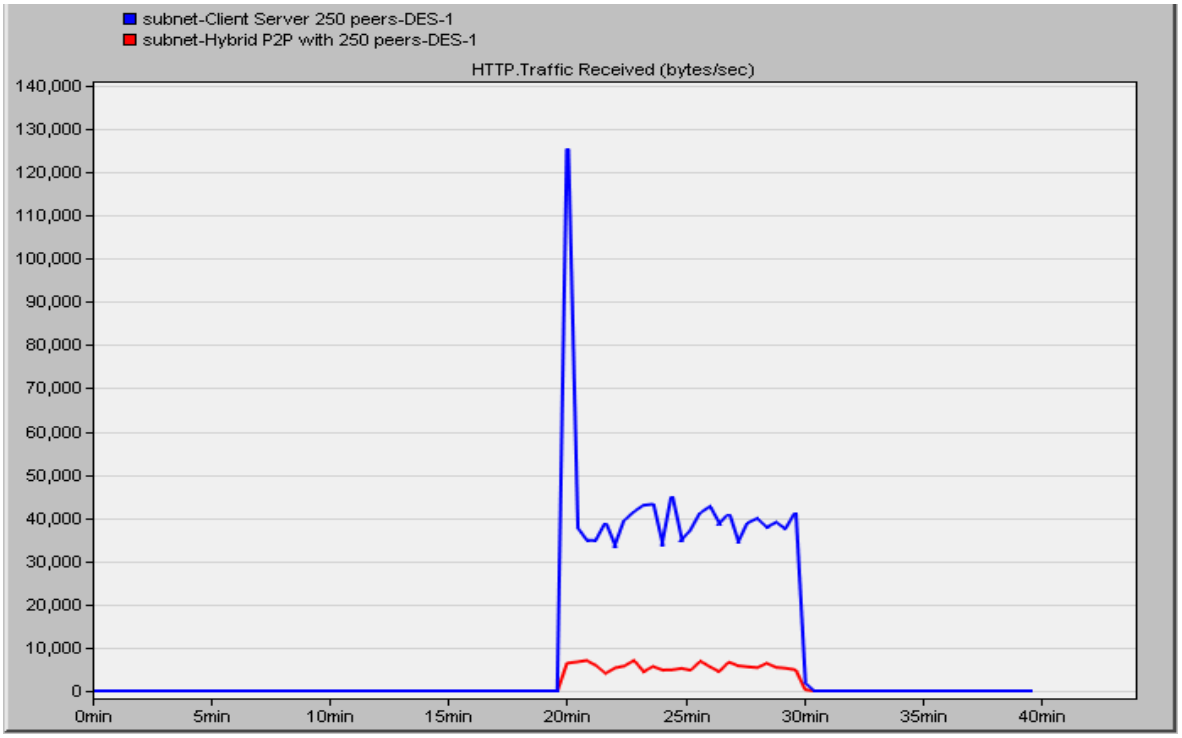


Figure 4. 24: The Traffic Received of HTTP for both MMOGs Hybrid P2P and Client/Server with 250 peers

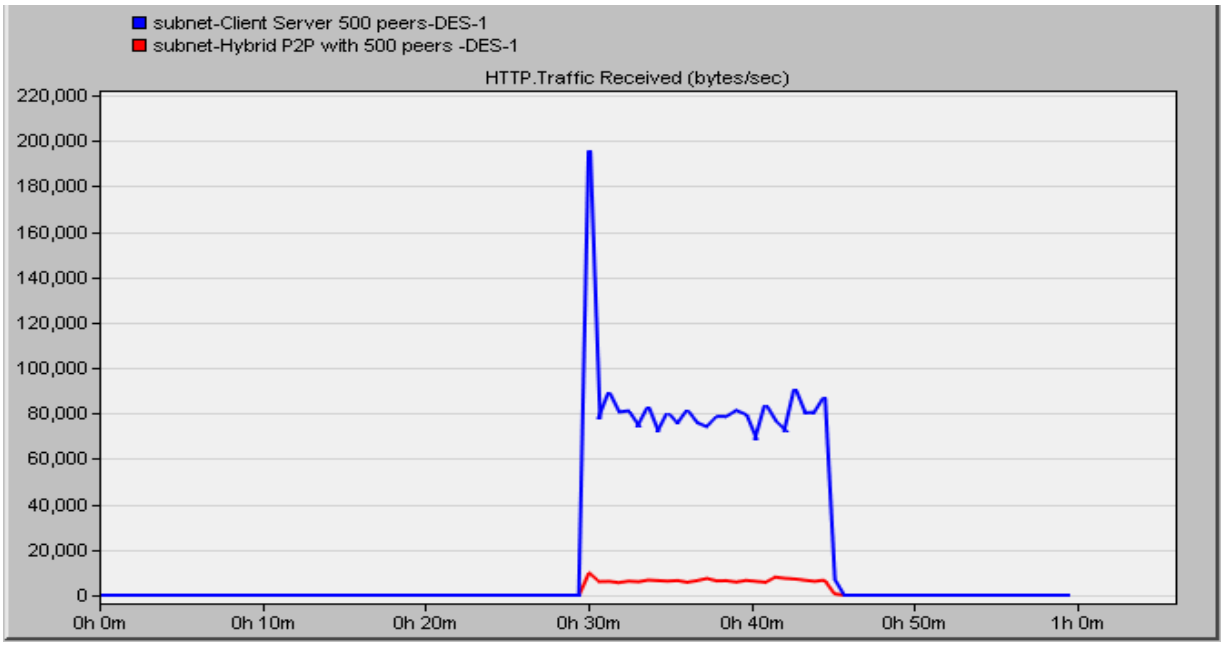


Figure 4. 25: The Traffic Received of HTTP for both MMOGs Hybrid P2P and Client/Server with 500 peers

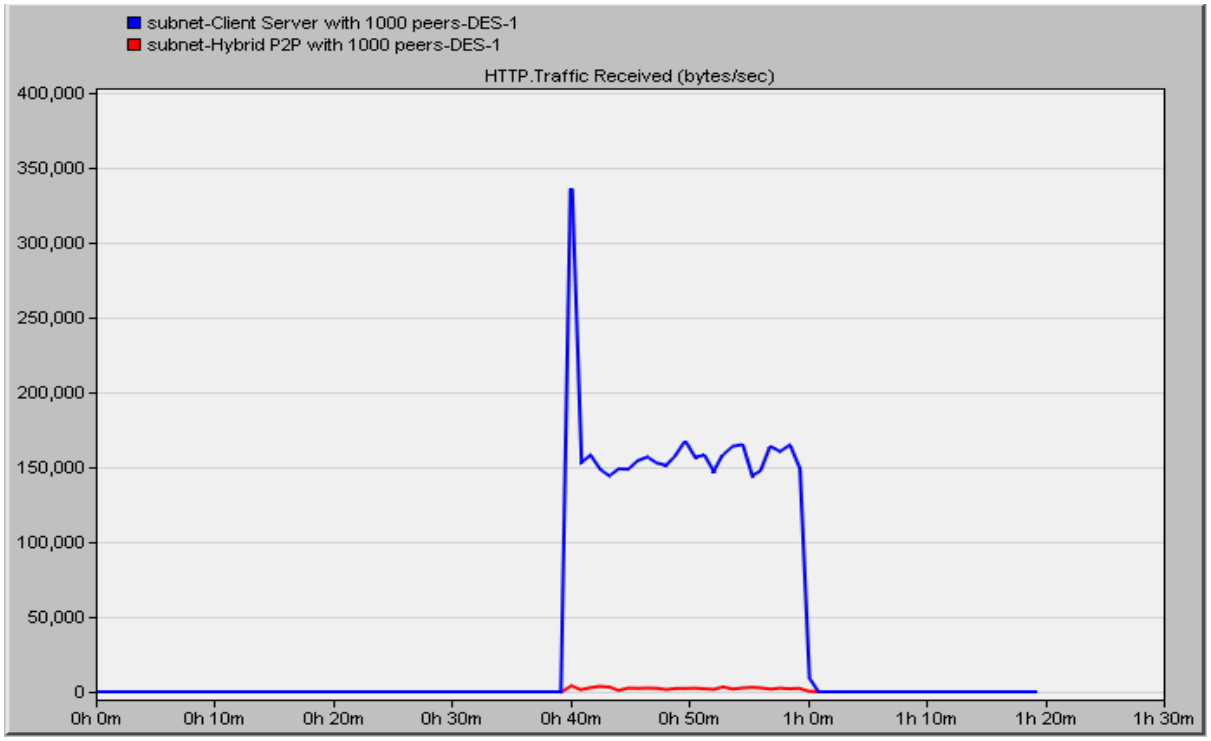


Figure 4. 26: The Traffic Received of HTTP for both MMOGs Hybrid P2P and Client/Server with 1000 peers

4.5.5.5 Traffic Received for FTP

File Transfer Protocol (FTP) has two primary operations for data transfer. The first one is FTP put operation uploads a file onto the FTP server; however the second is FTP get operation downloads a file from the FTP server onto the client node. Figures 4.27, 4.28, 4.29 and 4.30 show the FTP Traffic Received for client/server and hybrid P2P system. The figures indicate the big difference between the FTP traffic received for client server and the FTP traffic received for hybrid P2P system at the start time of the FTP application. The FTP traffic received for hybrid P2P is nearly equal to 50.000 bytes/second at the starting time of the application, while rapidly increasing even up to the highest level at the fourth scenario with nearly 340.000 bytes/second. This result will make the client/server architecture inappropriate to some extent to design MMOGs.

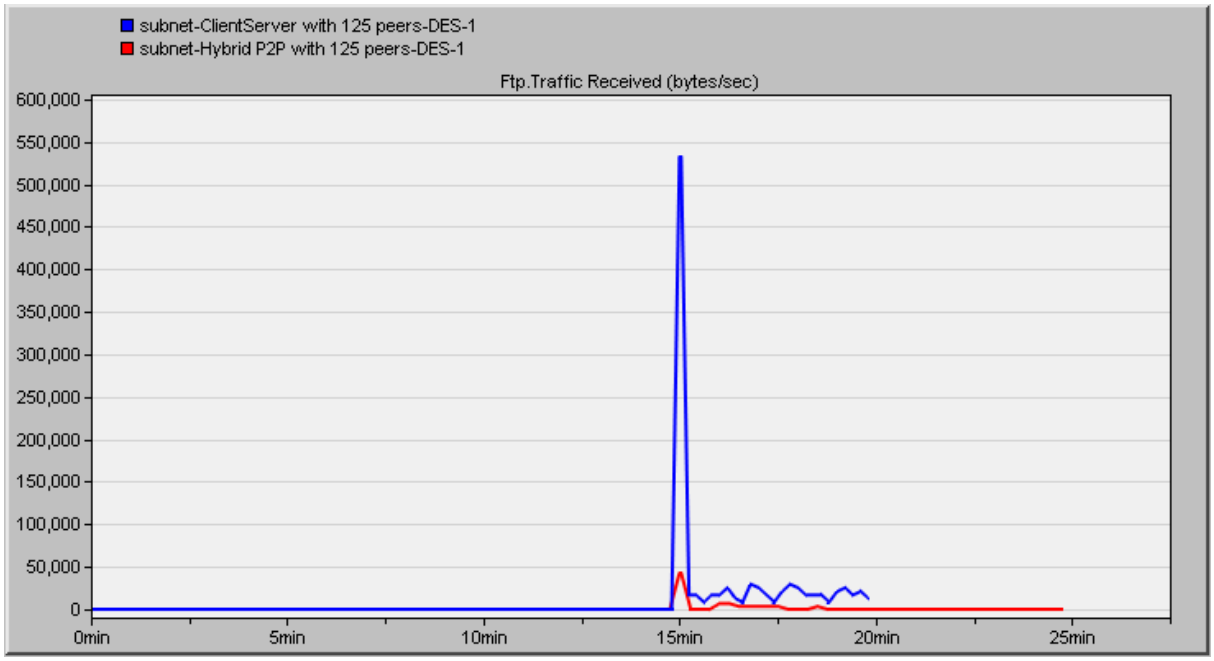


Figure 4. 27: The Traffic Received of FTP for both MMOGs Hybrid P2P and Client/Server with 125 peers

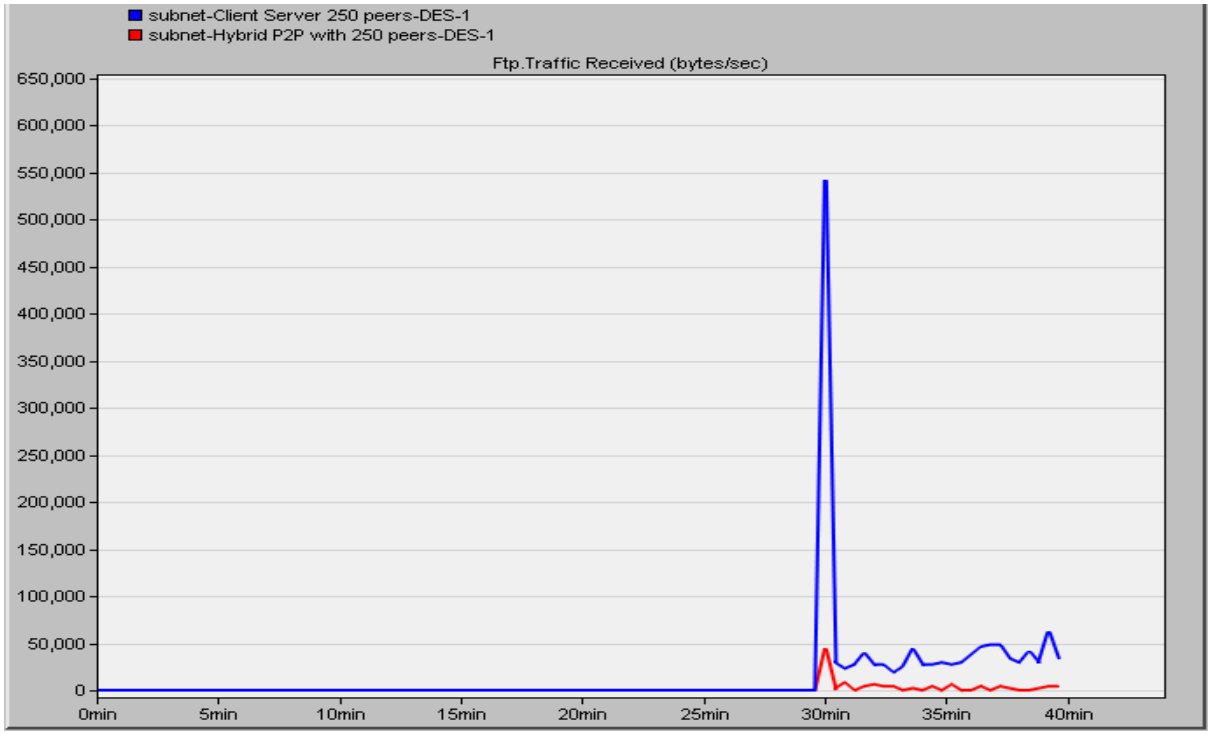


Figure 4. 28: The Traffic Received of FTP for both MMOGs Hybrid P2P and Client/Server with 250 peers

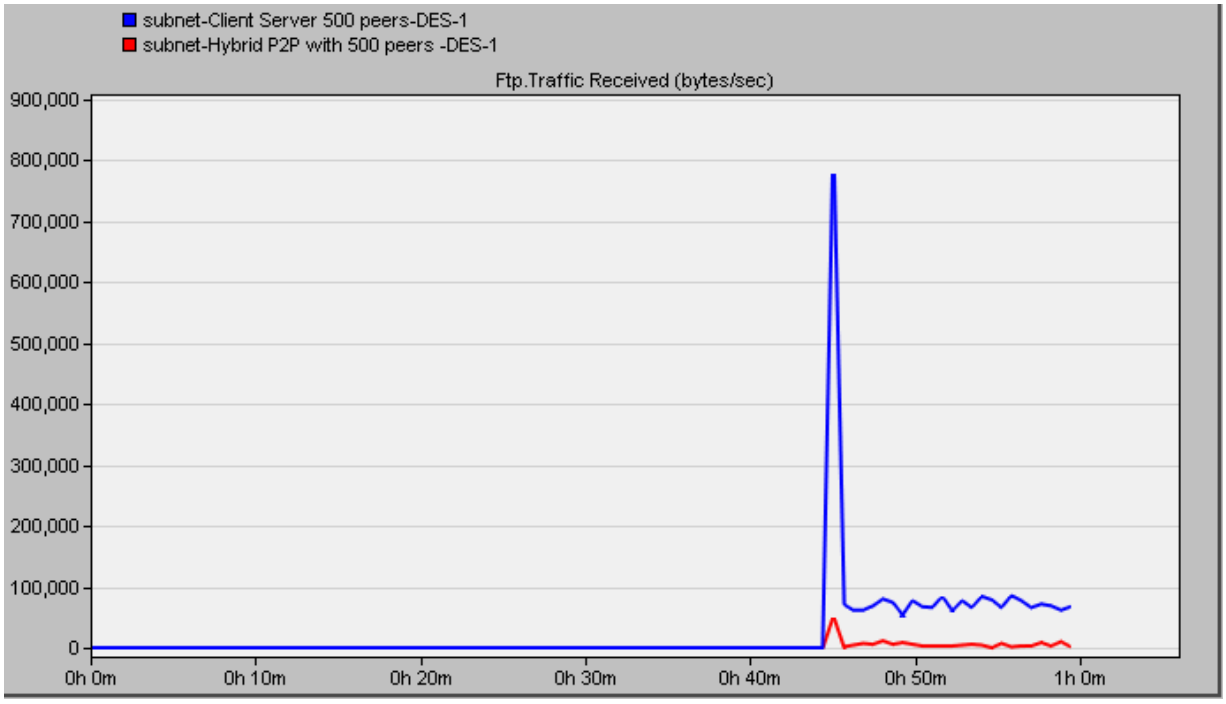


Figure 4. 29: The Traffic Received of FTP for both MMOGs Hybrid P2P and Client/Server with 500 peers

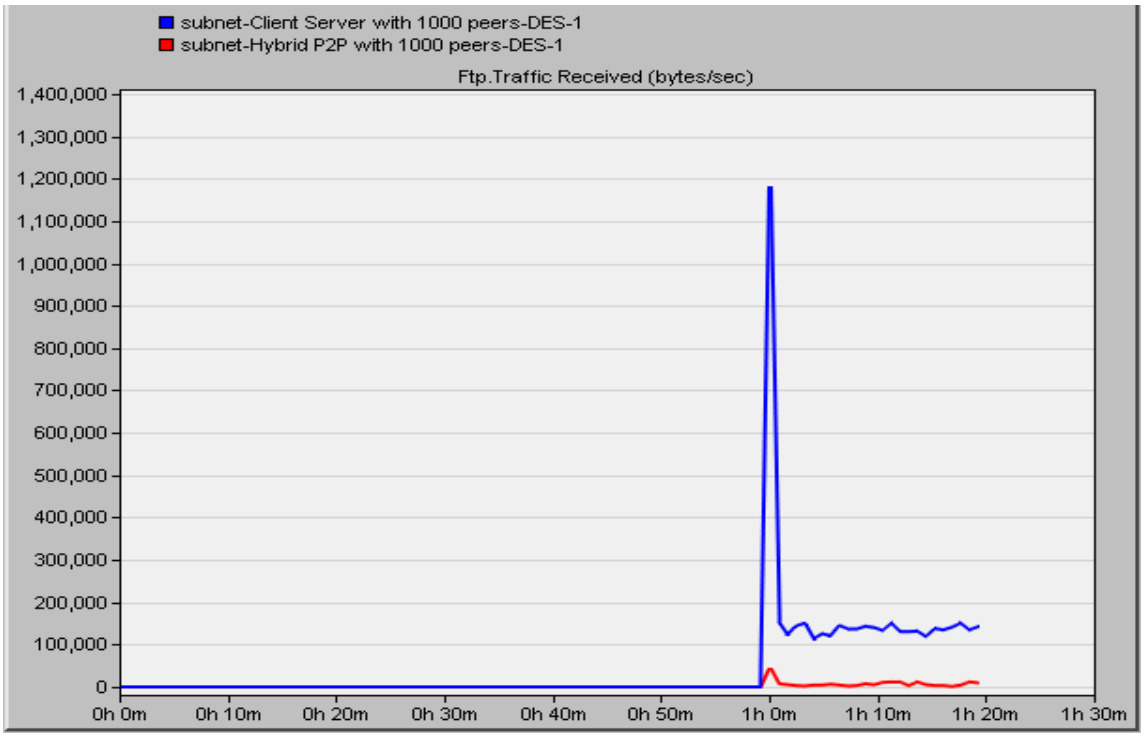


Figure 4. 30: The Traffic Received of FTP for both MMOGs Hybrid P2P and Client/Server with 1000 peers

4.6 Conclusion

The increasing number of players in MMOGs that support millions of players leads the game to several problems that relate to the scalability. In this chapter, we have discussed the design of MMOGs based on client/server and hybrid P2P architectures. We have proposed a new method to cope with the scalability problem for MMOGs based on hybrid P2P system. The systems have been simulated using OPNET 18.0 Modeler with different numbers of peers to validate the scalability of the new proposed architecture. The results have been compared with the client/server system to show the difference in delay and traffic received for various network applications, which are used in MMOGs for different players' administration and game state management, such as remote login, database, HTTP, and FTP. The results show that the hybrid P2P system is more scalable for MMOGs when compared with the client/server system.

5

SIMULATION OF MASSIVELY MULTIPLAYER ONLINE GAMES COMMUNICATION USING OPNET CUSTOM APPLICATION

5.1 Introduction

Online games generally involve a lot of actions that require fast response and low delay, and lost packet retransmission is impractical in these circumstances. Massively Multiplayer Online Games (MMOGs) are the type of online interactive game. As a result, traffic generated by online interactive games commonly consists of a huge number of UDP packets. The traffic is also highly periodic because this requires consistency and updating between the players and the game server states. One of the key important features in these games is the massive number of players whom play concurrently over the Internet. The huge number of players leads to more interactive, complex, and attractive game environment. The main reason that makes the popularity of MMOG's are increasing so rapidly in recent years is the experience of playing with other human players. With the increasing number of players in MMOGs, it became difficult to control efficiently the communication between the players and the game server, as well as between players. The standard application models of OPNET Modeler are completely easy to configure. They provide a sufficient level of detail for designing commonly used applications such as remote login, FTP, and HTTP. The standard applications just model a two-tier communication paradigm. It does not authorize for modification of the simulation application protocols. In addition, standard application in OPNET Modeler does not support the game communication, therefore, we have used the custom application to deal with this aspect of the traffic and carry out evaluation of the different architecture using different scenarios. In

order deal with this issue, OPNET Modeler supports significant facilities for modelling the custom applications. The custom applications are used to represent non-standard and multitier applications that follow a user defined protocol [74]. In the next section, we explain the concept of custom application of OPNET Modeler.

5.2 OPNET Custom Application

The custom application is an OPNET application modelling framework that is used to model a wide range of applications. It is used to create a new application model when the application of interest does not match to the standard applications. The custom application has enough attributes that enable you to configure different kinds of applications in detail. One of the most important advantages of custom application is modelling of complex multi-tier applications. In addition, a custom application is used to represent any number of tiers.

5.2.1 Custom Application Architecture

In order to generate network and server traffic, we have programmed the Custom Application according to specified patterns. The Custom Application has the following features:

- There are more than two hosts that can be involved in processing and data transfers. The sequence of hosts can be specified in each task. The sequence can start with a server or workstation, end with a server or workstation, and can use any group of servers or workstations in between.
- Hybrid P2P interaction is task-based. A task consists of several interactions between a server and peers or between super-peers.

The following components are used within the custom application [75]:

- **Task**: A basic unit of user activity within the context of an application. The entire task is totally complete just when the last phase of the task has been finished. For example: reading an e-mail message, obtaining records from a database or sending a query to the server. The start and end of a task must be clearly defined.
- **Phase**: An interval of related activity that is contained in a task. Each phase consists of a processing event and/or a data transfer, which can happen in any end device. This end device will become a tier for the application. Subsequent phases are usually assigned to occur in a chain. Therefore, the destination of the data transfer phase will become the source of the next data transfer phase

- **Step**: A phase consists of many steps, for example obtaining a simple HTTP browsing from a web server involves two steps: first, sending a request to the web server, and second, receiving the corresponding results from the web server.
- **Tier**: A single software process that is used for executing some or all of the steps in a task.

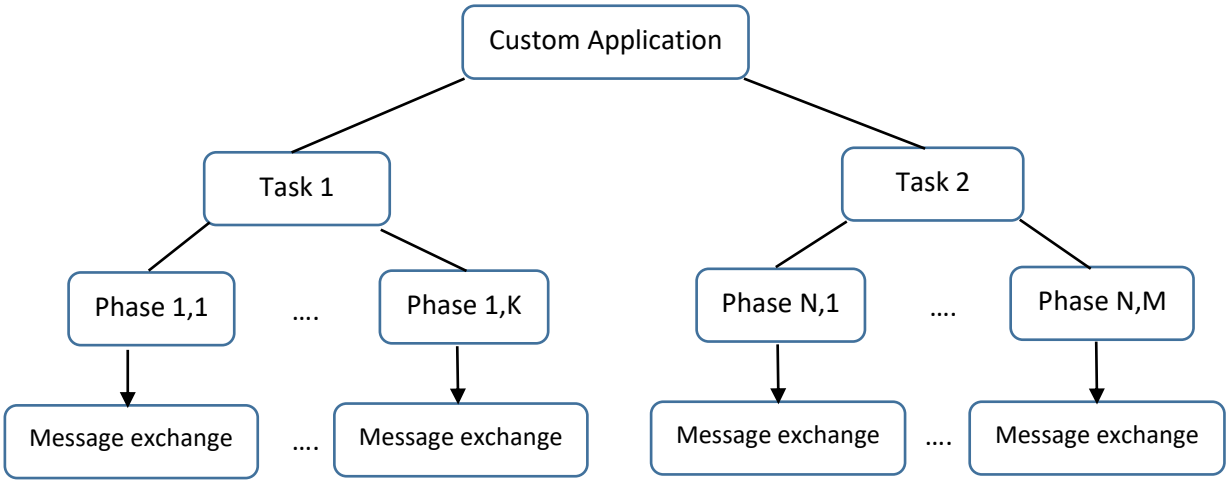


Figure 5. 1: Custom Application Hierarchy

5.2.2 Custom Application Workflow

There are some steps we must take into account when modelling the custom applications. These steps are listed below [74]:

1- Define the application that is used in the network topology and all the steps in the transactions. The transaction steps determine the tiers, determine the task phases, breakdown each phase, and identify the parameters.
2- Determine all the Phases of a Task.
3- Configure all the Tasks in the Task definition node.
4- Configure the custom application, choosing "Custom," instead of any standard applications, and listing the tasks that form this application.
5- Define user profiles, by using Profiles definition node.

5.3 Design of OPNET Custom Application Simulation without Players Movements

Game simulation is widely used to evaluate new architectures and protocols. The game communication is considered one of the most important aspects in the game simulation. We have used the OPNET Modeler custom application to present a new technique to communicate between players and game server in the game world. OPNET Modeler 18.0 have been used to

implement and simulate our MMOGs based on hybrid P2P architecture and compared the results with the traditional MMOGs Client-Server architecture. Furthermore, we have compared the transport protocol TCP versus UDP for each scenario in the project. The network topology consists of three scenarios for each architecture which will be explained in the following sections.

5.3.1 Design of MMOGs Client-Server Architecture based on Custom Applications

The vast majority of the MMOGs have been developed using the classic client-server architecture. This architecture demonstrates some problems such as all the load of handling the virtual world is carried by the server. The players connect directly to the server through the client application on their own computer and the server is responsible for handling all the rules and the state of the virtual world. One of the first things we have to be aware of is that due to the big number of clients that will connect to the server as well as the massive load that they create it will be impossible for one single server machine to handle everything. A lot of effort is put into developing better and more effective ways to deploy the load over multiple servers. This method can achieve the largest possible number of the participating players and experience with a minimum of needed resources. One of the most common optimizations used by game communication is area of interest (AoIM). Individual players usually only interact with a small area of the game world at any one time. Servers only need to update clients about objects in this AoI. This leads to a reduction in the number of objects transferred from the total set of objects to the number of objects in this area [11].

The MMOGs Client-Server architecture consists of the centralised game server and several clients (players). The players are directly connected to the game server. Using OPNET Modeler 18.0 to design the network topology for this architecture. The following devices are used for the design: game server, two Cisco C400 Router, IP Cloud, Ethernet Switch, and Ethernet Workstations as shown in figure 5.2. Ethernet Workstations are used to represent the clients (players) and are grouped using Ethernet Switch through the use of Ethernet 100BaseT cable. The group of players are directly connected to the game server by using three devices: Cisco C400 Router (Client gateway), IP Cloud (internet), and Cisco C400 Router (server gateway). The number of players are 125, 500, and 1,000 for each scenario. The game applications are controlled and managed by the central server. We use task configuration to configure the custom application for this scenario.

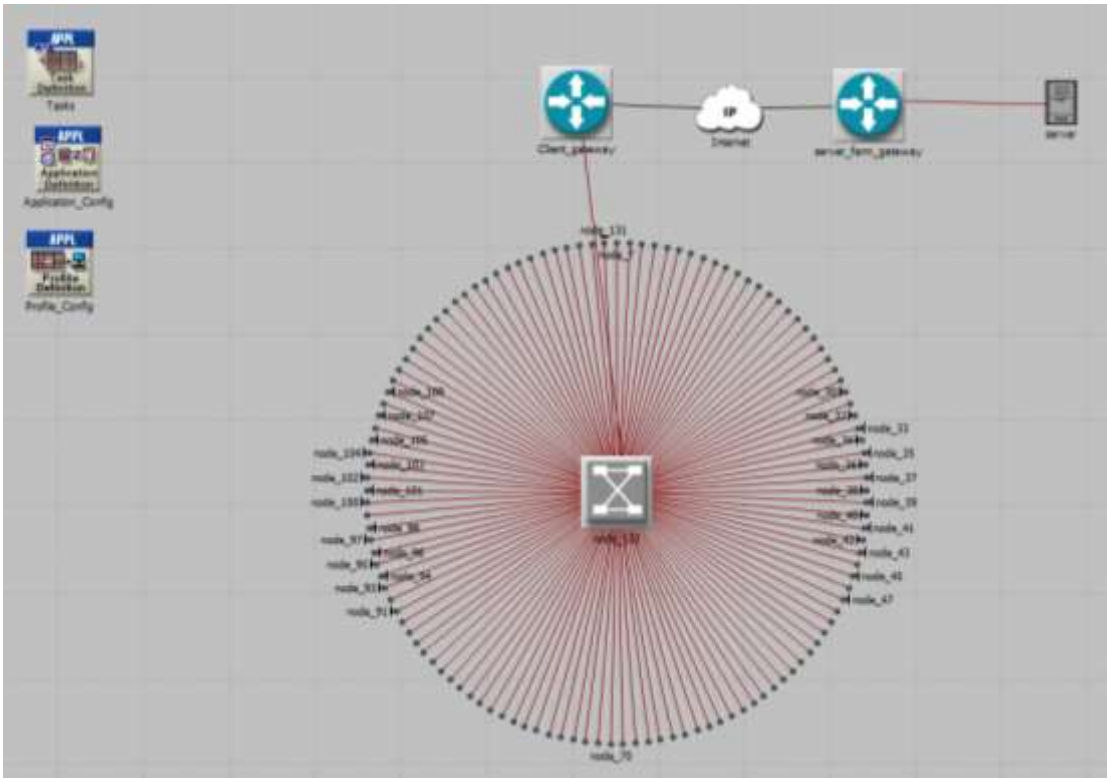


Figure 5. 2: MMOGs Client/Server Architecture

5.3.2 Design of MMOGs Hybrid P2P Architecture based on Custom Applications

The primary requirement for MMOGs based on Peer-to-Peer architecture is to maintain state consistency, and a shared sense of virtual space among great numbers of players without the need for support by the server [76]. The development of peer-to-peer architecture for MMOGs is a significant research topic. There are great deal of researches which have been done on the use of peer to peer architectures to design MMOGs, but until now this has not led to it being used in any commercial MMOGs. Because the use of P2P system to design and develop multiplayer games is a new trend and it hasn't reached the mainstream consumer market. P2P architecture allows clients to communicate directly with each other and they are responsible for a small portion of the game state, and therefore do not need a central game server. Removing the need for a central game server will decrease the cost of running a game dramatically. However, there are a lot of problems when designing and developing a peer-to-peer architecture for MMOGs. For example, a major problem will be to save the game state, the deployment of game updates will be more troublesome, and the bandwidth used for the clients will be extremely high when compared with the client-server architecture. Peer-to-peer architectures do not have a central server and therefore it becomes more difficult to present a single consistent virtual world to all users. In the Peer-to-Peer system, distributing the game

state over all the peers in the game world, there will have to be an effective way to set different elements of the game world to different peers [76].

The MMOGs hybrid P2P architecture consists of the centralised game server, super-peer, clone-super-peer, and several peers (players). The players are grouped into regions and each region has 60 peers. The players in the region are directly connected to both super-peer and clone-super-peer. The hybrid P2P scenario has the same devices in section 5.3.1 but with change in the zone area by adding super-peer and clone-super-peer to the region as shown in the figure 5.3. The game applications are controlled and managed by the region's super-peer and clone-super-peer. The super-peers and clone-super-peers are connected to the game server. The game updates are sent from the server to the super-peer and clone-super-peers of each region. We have used task configuration to manually configure the custom game application for MMOGs based on hybrid P2P system.

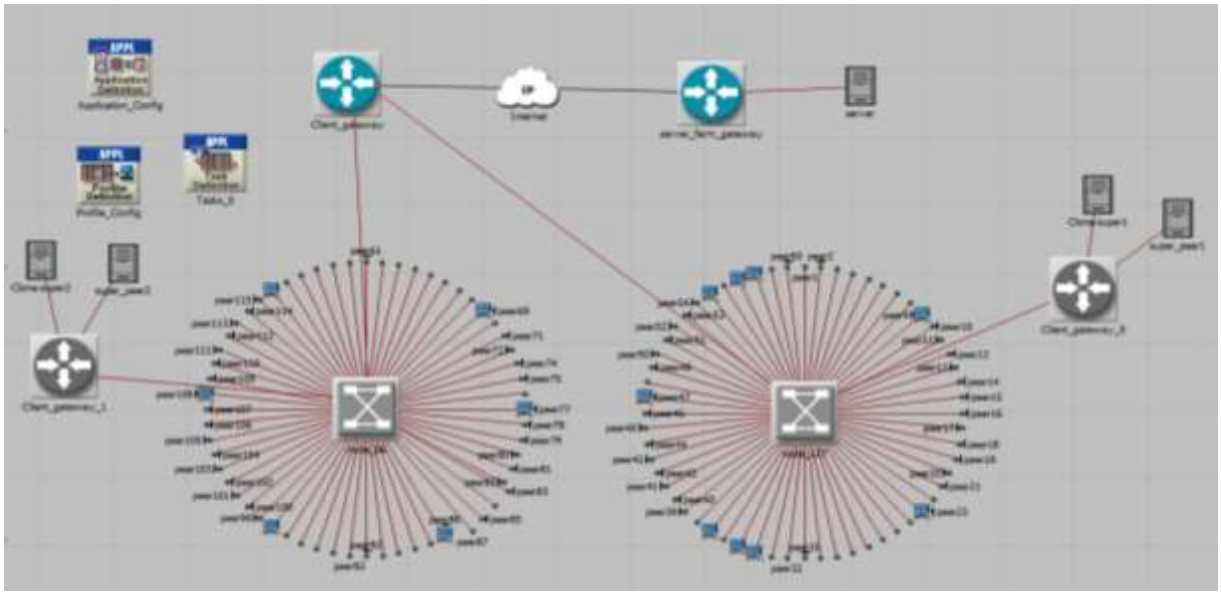


Figure 5. 3: MMOGs Hybrid P2P Architecture

5.4 Custom Application for Game Communication

As the hybrid P2P architecture is not currently supported in OPNET Modeler, or any other simulator, we have created a custom application for game communication between client and server in client-server architecture, and between peers, super-peers, clone-super-peers, and server in hybrid P2P architecture. The custom application for client-server architecture consists of two phases. The first phase is used to communicate between clients and server, however the second phase is used to communicate between server and clients. While, the custom application for hybrid P2P architecture consists of 20 phases for the scenario with 125 peers, and the

number of phases will increase whenever the number of peers is increasing till up to 80 phases for the scenario with 1,000 peers and 16 regions. The number of regions will increase with the increase in the number of peers. Figures 5.4 and 5.5 show the custom application for client-server architecture. In figures 5.4 and 5.5, the phase 1 is used to communicate clients with the server, however the phase 2 is used to communicate the server with clients. Figures 5.6 and 5.7 show the custom application phases for a hybrid P2P system that is used to communicate between the peers and both regions super-peer and clone-super-peer, as well as the communication between server, super-peer and clone-super-peer. The OPNET Modeler custom application provides some advantages for hybrid P2P architecture. These advantages are game communication, boot strapping, management of the login and leaving of player, controlling the migration from one region to another, easy management of the increase of players, and management of the failure and recovery.

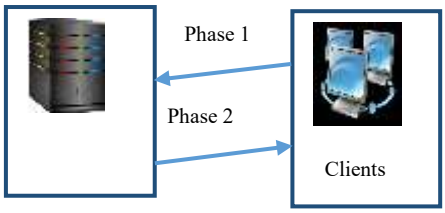


Figure 5. 4: Custom Application Phases in Client-Server System

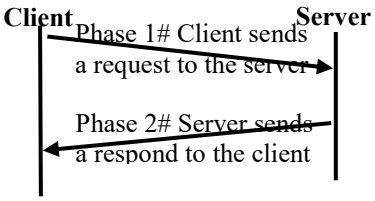


Figure 5. 5: Phases Description in Client-Server System

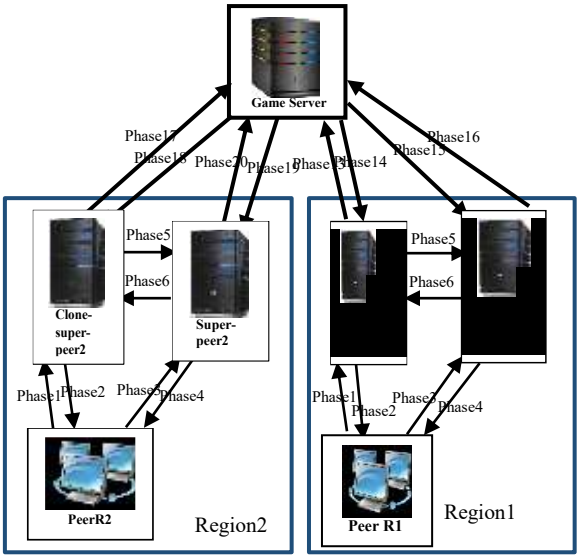


Figure 5. 6: Custom Application phases in Hybrid P2P System

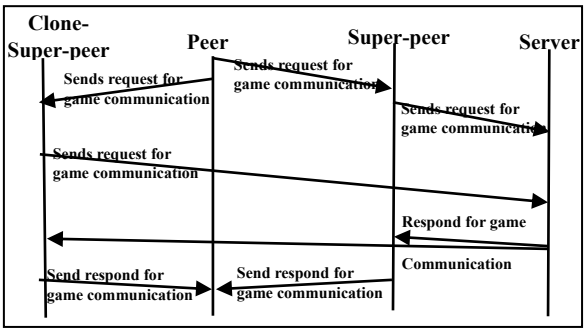


Figure 5. 7: Phases Description in Hybrid P2P System

5.5 Reducing the Traffic volume

Reducing traffic of the network has been concentrated on two prime dimensions: Reducing the number of parties (peers, super-peer, or clone-super-peer) that are sent a message to, and reducing the number of messages required to be sent from the server to the parties in the game world. MMOGs are a group of state information, and a process for updating those states. There are many state changes when two avatars are interacting with each other. The game server can be changing just the state of the two avatars. There are three key solutions to exploit this behaviour: region-based computation, message deployment, and Area of Interest Management (AoIM). In region-based mechanism, the game world is divided into regions. In our research, we use the geographical based region to divide the game world, and each region manages the state and controls the behaviour of players by super-peer and clone-super-peer. The message deployment and game states are done through both super-peer and clone-super-peer. This mechanism will reduce the network traffic compared with sending the message and

game states to each player in the game world. However, in Area of Interest Management (AoIM) mechanism, a message is only sent to the player that needs this information, in order to display an accurate state of the world to the players. In this research, two types of traffic are used: explicit traffic and background traffic as detailed in the next subsections.

5.5.1 Explicit traffic

Explicit traffic is on a packet-by-packet basis traffic. It models packet creation, queuing, transmission, and deletion explicitly through a discrete-event simulation process. The packet is created by using several applications such as (remote login, database, web services, HTTP etc.). Therefore, explicit traffic modelling is one of the most important models that provide accurate results because it provides the details of protocols. In addition, explicit traffic can be used to model all the details in discrete-event simulation, but it takes more time and resources. Using background traffic in order to reduce the computational burden, this will be introduced in the next section. There are three main mechanisms for explicit traffic modelling [75]:

- Packet generation: The basic method to generate streams of generic packets is configuring certain node objects.
- Application requests: Create application requests to represent the traffic flowing among two nodes in the network.
- Application traffic models: Include a group of models for generating traffic based on standard applications such as (remote login, database, web services, HTTP etc.).

5.5.2 Background Traffic

Explicit traffic provides an accurate result but it is not useful for large networks because it allows detailed control of the modelled protocol and takes a very long time and requires further computing resources. Therefore, use of background traffic has been introduced to solve these problems. Background traffic is described as analytically modelled traffic rather than discrete-event simulation. The main purpose of using background traffic is to model the impact of general traffic in the network on a chosen traffic of interest. Background traffic causes further delay and response times for explicitly modelled traffic. Therefore, it consumes a very small amount of computing resources and takes considerably less time. Hybrid simulation system is used to combine explicit traffic and background traffic. The performance of explicit traffic in this case is really influenced by the extra delays analytically calculated and produced by the background traffic load. However, background traffic technique is not beneficial for all

simulation scenarios because it is based on analytical models. The implementation of background traffic is restricted to statistics of protocols at higher layers than IP, such as TCP, UDP, IP traffic, and applications. Background traffic can takes three different forms [74]:

- Traffic flows: Describes an end-to-end flow of traffic from a source to one or more destination nodes. In this type of traffic, you can create traffic flows manually by using traffic flow objects. Also, you can import traffic flows from text files and spreadsheets.
- Baseline loads or sometime called "static background utilization": The traffic is represented as a background load on a link, node, and connection. In this type of traffic, a traffic load is "static" and applies to one object, on the contrary a traffic flow can extend to multiple nodes and links.
- Application demands: Use application demands to represent background traffic flowing between two nodes. Furthermore, you can also configure application demands to be purely discrete- event simulation (explicit traffic), or hybrid simulation.

5.6 Design of OPNET Custom Application Simulation with Players Movements

The main goal of game simulation is to obtain a situation with conditions approaching the real environment. The simulation results obtained are very precise and more related to the behaviour of the real environment. One of the most significant issues of game networks is the ability to move from one region to another, inside the game world. The possibility of controlling the movement of player nodes allows more efficient prediction and organising of network sources for individual stations, for example handover optimization. In OPNET Modeler environment, there are several types of trajectory [77] explained in the next section.

5.6.1 Segment-Based Trajectories

A segment-based trajectory consists of one or more traversal-time values that define the node site's movement and group of a three-dimensional path (X, Y, altitude) that define the orientation in the game world. These trajectories are stored in ASCII text files with a '.trj' extension and are located to a player node or subnet by using the "trajectory" attribute. A player node follows its trajectory by moving during simulation in a great-circle path from one point to the other. A node's position is specified at a given time by interpolating between the trajectory points promptly before and after that time. There are two categories of segment-based trajectories fixed-interval and variable-interval. In this trajectory, just one value is located for the traversal time of all segments. Therefore a node takes the similar quantity of

time to traverse every segment, regardless of the length of segment. However, in variable-interval trajectory, each point has its own particular altitude, such as, wait time, orientation, and traversal time (the time from the previous point to the present point). The waiting time causes a node object to pause at each point before it starts traversing the next segment [77].

5.6.2 Random trajectory

We can use the random mobility feature to model random movement. Random mobility is used to define a rectangular region in which a site will move during a simulation. The region is defined by specifying x-y coordinates or by using a mobility domain (a special attribute in wireless domain). During simulation, the site randomly chooses a destination in the region and moves toward it at a particular or randomly selected speed. Upon arriving at its destination, the site stops a configurable length of time before it reiterates the process by choosing another random destination. A mobility configuration object contains random mobility parameters as profiles in order to make it easy to reuse them [77]. In our research simulation, random node site movement is used to allow the players to move from one region to another or even in the same region.

5.6.3 Direct Manipulation of Position Attributes

OPNET Modeler has the possibility to model movement of mobile sites by using directly manipulating position attributes. The path of mobile site is predefined for the whole simulation, if a trajectory is located for that site. While, if there is no specific trajectory, the site's location can be immediately updated by any process during simulation. A node site's x and y position attributes determine its location in its parent subnetwork. However, a mobile site's altitude attribute determines its elevation relative to sea level and the underlying terrain. Any change in one of these attributes will cause a direct change in the node site location. There are two techniques used to dynamically change the node site location. The main issue for the simulation should be explained before the simulation results. The first approach is a centralised technique. In this approach, one process is responsible for updating the positions of all of the node sites in a network model. However, the second approach is a decentralised technique. In this approach, each node site has a process implementing within it that updates just its own position [77].

5.7 Design of MMOGs Client-Server Architecture based on Custom Applications with Players Movement

We have used custom application to design MMOGs client-server system to allow the players to move among the regions or move inside the player's region. The architecture is designed similar to the design in section 5.3.1 but with using wireless devices to represent the players in the game world. Wireless workstation is used instead of Ethernet workstation in order to provide a flexible movement of players inside the game world. Figure 5.8 shows the design of MMOGs client-server architecture based on custom application with players' movement.

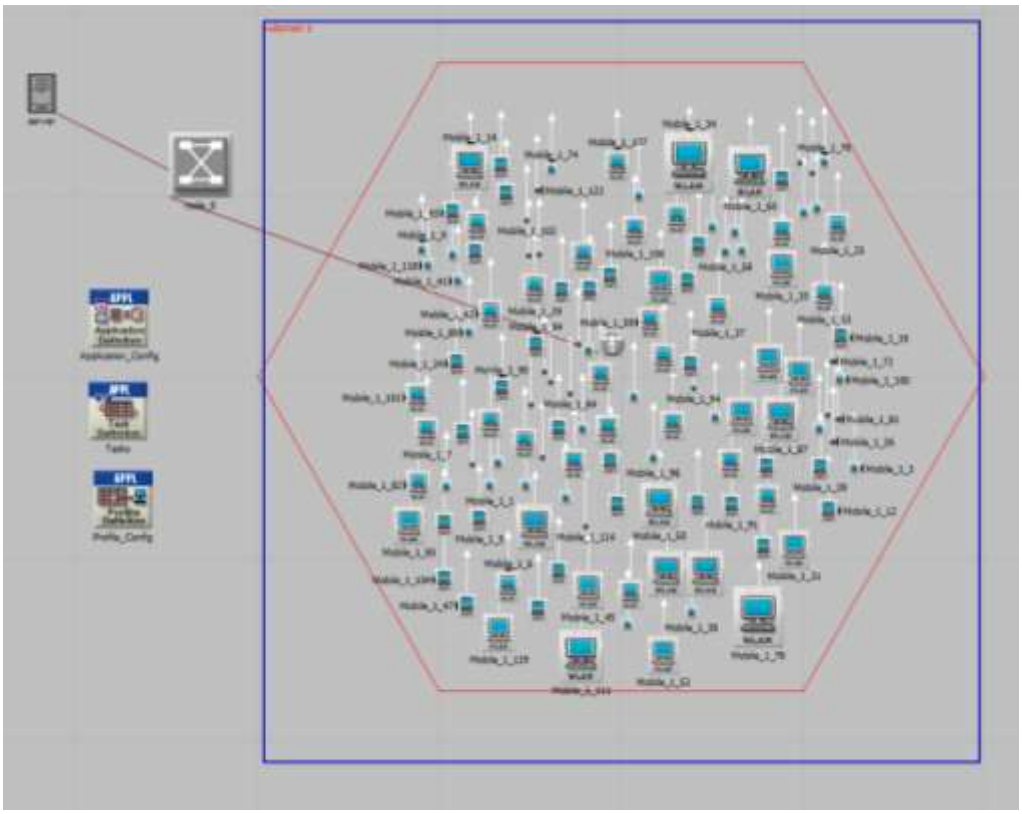


Figure 5. 8: MMOGs Client-Server based on Custom Application with Players' Movement

5.8 Design of MMOGs Hybrid P2P Architecture based on Custom Applications with Players Movement

Overall, in order to design MMOGs more close to the real world, we have designed MMOGs hybrid P2P based on custom application with players' movement from one region to another or even in the same region of the game world. The network design of this architecture is similar to the design in section 5.3.2. However, we have used a wireless workstation to

represent the players in the game world. A wireless workstation provide a high level of elasticity to allow the player for moving inside the region, as well as moving across the regions. We have used the region border to separate each region inside the game world. Figure 5.9 illustrates the design of MMOGs hybrid P2P architecture based on custom application with the player's movement.

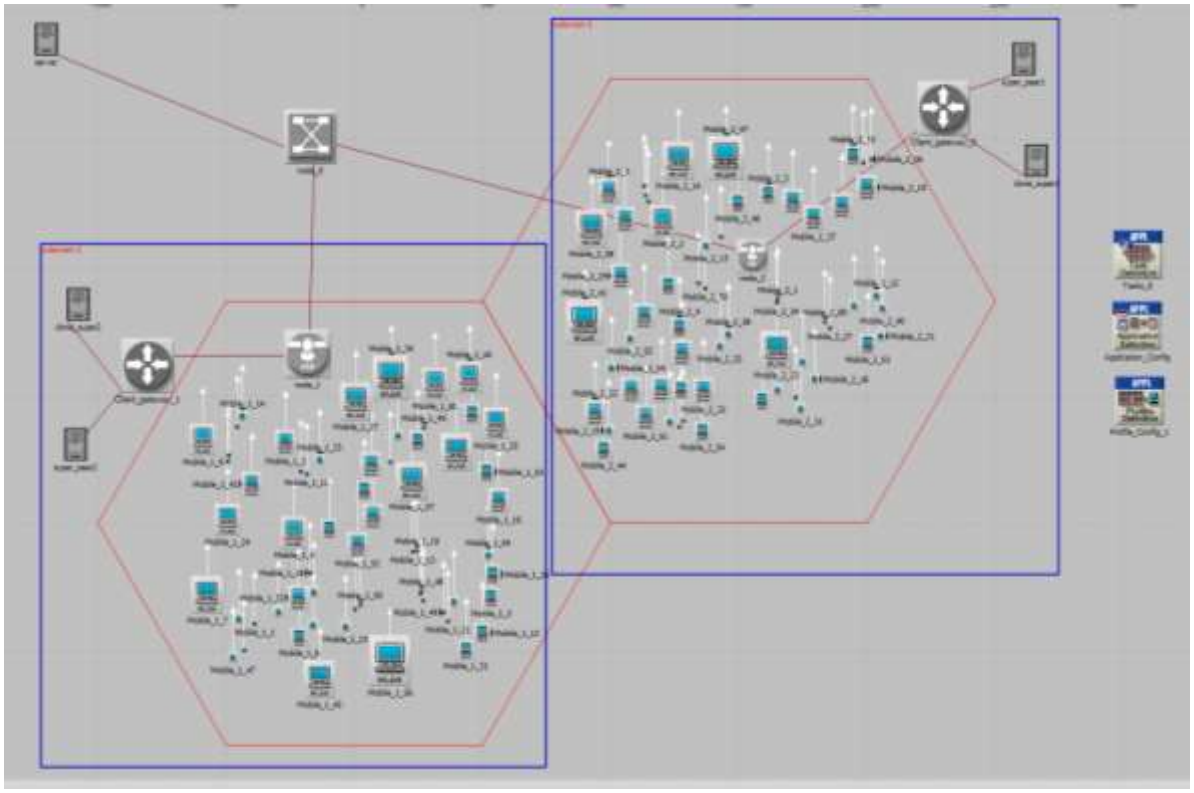


Figure 5. 9: MMOGs Hybrid P2P based on Custom Application with Players' Movement

5.9 Simulation Results

There are two parts of the simulation results: simulation results for OPNET custom application and simulation results for OPNET custom application with player's movement. These types are explained in the following sections.

5.9.1 Simulation Results for OPNET Custom Application

There are five main simulation results that were collected after the simulation was completed. These results are introduced in the following sections.

5.9.2.1 Simulation Time

Simulation time is considered as one of the significant issues in the simulation. There are two concepts that help to understand the simulation time issue. The first concept is simulation duration. Simulation duration referring to the amount of time simulated. It is not the amount of clock time that it takes to complete the simulation. Specify the duration of a simulation is commonly driven by one of two factors. The first factor is how much of the activity will be simulated to get a valid result. The second factor is the time required for each activity. However, the second concept is called elapsed time. Elapsed time refers to the amount of real time that take to complete the simulation. The use of OPNET Modeler custom application reduces the elapsed simulation time for all the scenarios, due to the ability to control and manage the simulation attributes and traffic compared to the standard OPNET application models.

5.9.2.2 Overall Delay

Overall delay is considered one of the most significant issues in the networked game. This parameter is defined as the overall end-to-end delay for all packets received by the station. End-to-end delay for the application used during the simulation is measured from the time from source to destination. End-to-end delay refers to the time it takes to send a packet from source to destination over a network. Figures 5.10, 5.11, 5.12, and 5.13 show the overall delay for the MMOGs hybrid P2P architecture compared to the results of the overall delay for MMOGs client-server architecture using TCP and UDP transport protocol without adding any background traffic. Figure 5.10 illustrates a variation of delay when using hybrid P2P system for game communication compared with client-server system using TCP transport protocol. The client-server delay curve begin with nearly 0.0024 second and starts to decrease till 0.0008 second to be stable till the end of simulation; however the curve of hybrid P2P is more stable beginning with 0.0004 second and remaining stable till the simulation is completed. Figure 5.11 shows the overall delay for the same scenario but using UDP transport protocol. The figure shows the big variation of delay compared to the same scenario using TCP transport protocol. However, this variation is increased when the number of the players is equal to 500 as shown in the figure 5.12. In figures 5.10, 5.12 and 5.13 the delay graphs of MMOGs based on hybrid P2P system are nearly the same shape in all the three scenario, while the delay graphs of MMOGs based on client-server are be variation. Because of the game updates and the players' communication, hybrid P2P architecture requires less client management and control compared

to the client-server system. Consequently, MMOGs hybrid P2P architecture based on custom application produces less delay compare to the MMOGs client-server architecture based on custom application.

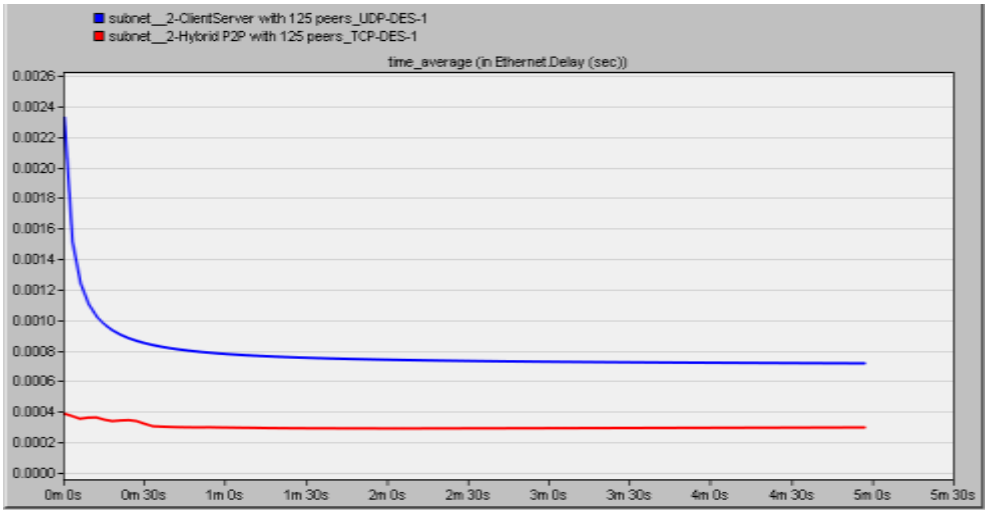


Figure 5. 10: Overall Delay for Hybrid P2P and Client/Server with 125 peers TCP Protocol without Background Traffic

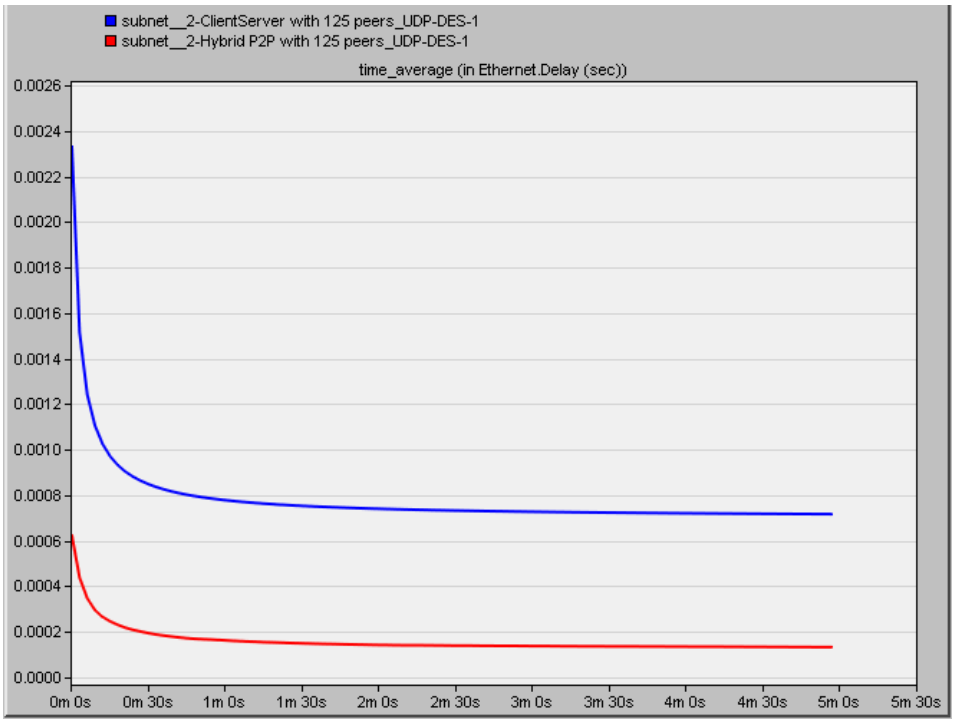


Figure 5. 11: Overall Delay for Hybrid P2P and Client/Server with 125 peers UDP Protocol without Background Traffic

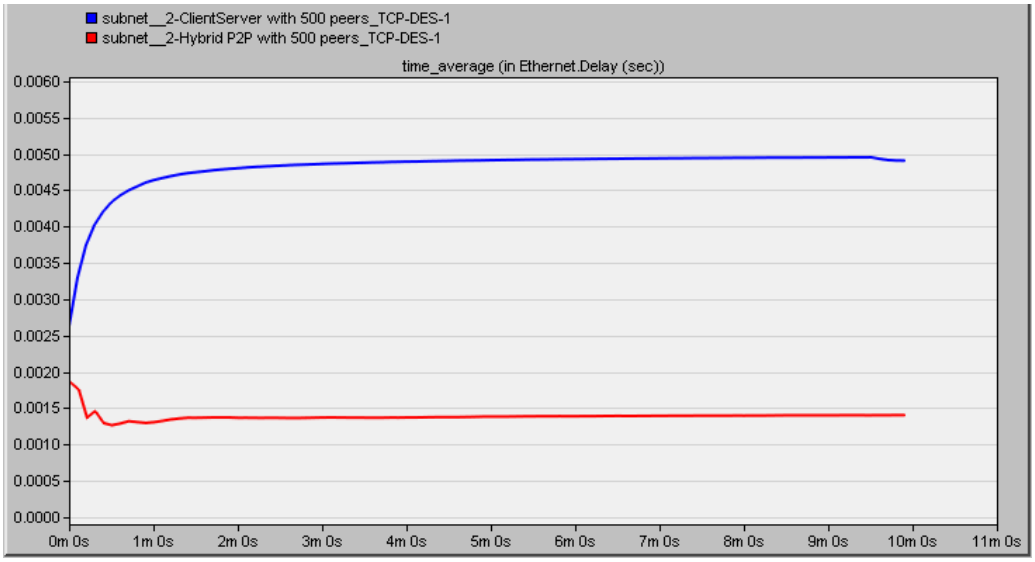


Figure 5. 12: Overall Delay for Hybrid P2P and Client/Server with 500 peers TCP Protocol without Background Traffic

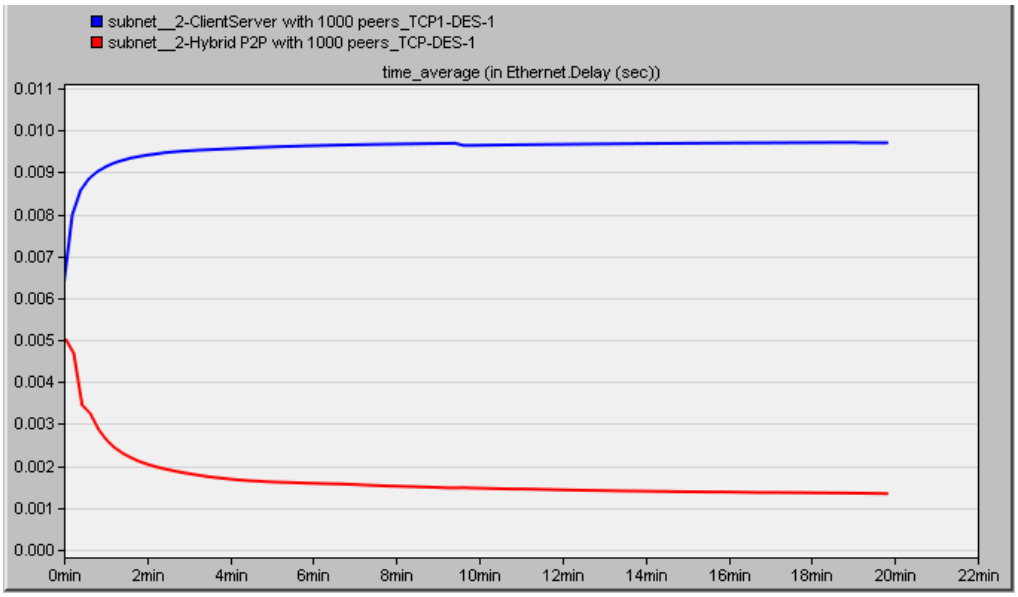


Figure 5. 13: The overall Delay for Hybrid P2P and Client/Server with 1000 peers TCP Protocol without Background Traffic

5.9.2.3 Traffic Received

Traffic received is the average number of bytes per second received by all the nodes in the network. In other words, network traffic is the amount of data moving across a network at a certain point of time. Network traffic is one of the main ingredients for measuring network traffic, network traffic control and simulation. The proper regulation for network traffic helps

to ensure the quality of service for the network. Figures 5.14, 5.15, 5.16, and 5.17 show the custom application traffic received for Hybrid P2P and Client/Server system using TCP and UDP transport protocol respectively without adding any background traffic. The figures below elucidate the great difference in traffic received for MMOGs hybrid P2P architecture based on custom application in comparison with MMOGs client-server architecture based on custom application using both TCP and UDP transport protocol. Figure 5.14 shows traffic received of MMOGs hybrid P2P system begin with nearly zero and slightly increases till arriving at 50.000 bytes/sec in the last minute of the simulation. However, the variation is less when using UDP transport protocol as shown in figure 5.15. As a result, the architecture of MMOGs hybrid P2P based on custom application produces traffic received considerably less than the traffic received of MMOGs client-server architecture based on custom application.

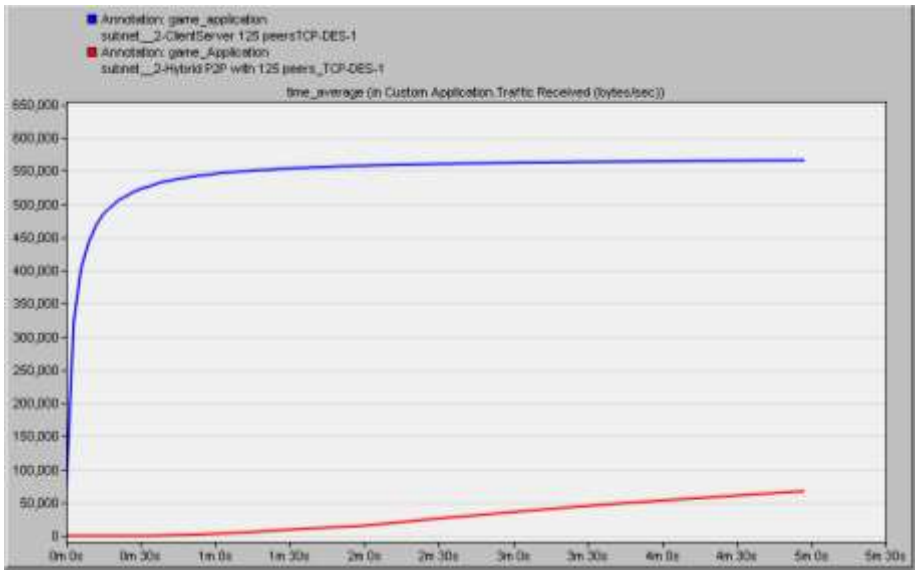


Figure 5. 14: Traffic Received for Hybrid P2P and Client/Server with 125 peers TCP Protocol without Background Traffic

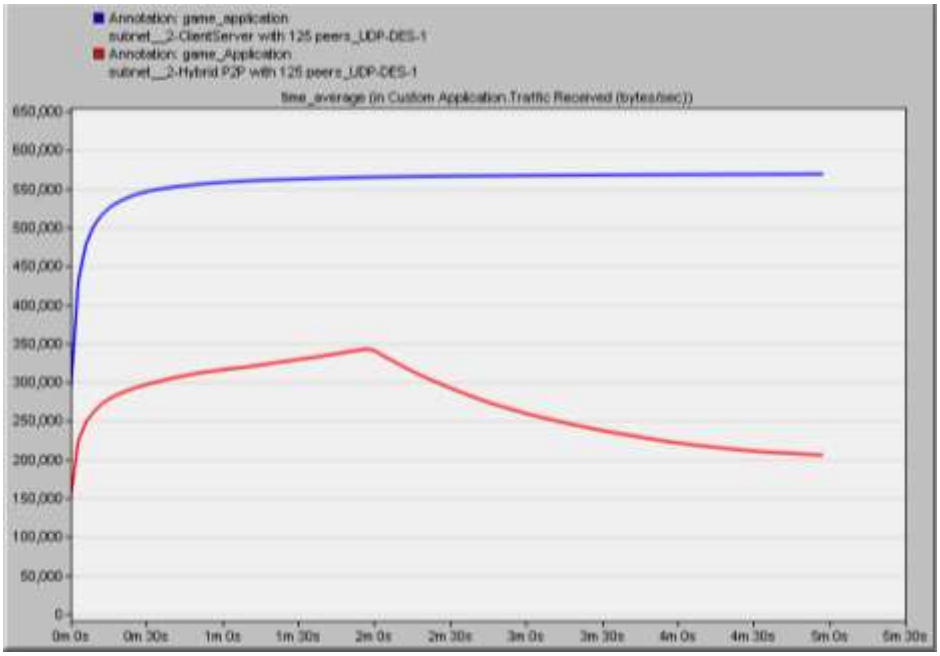


Figure 5. 15: Traffic Received for Client/Server and Hybrid P2P with 125 peers UDP Protocol without Background Traffic

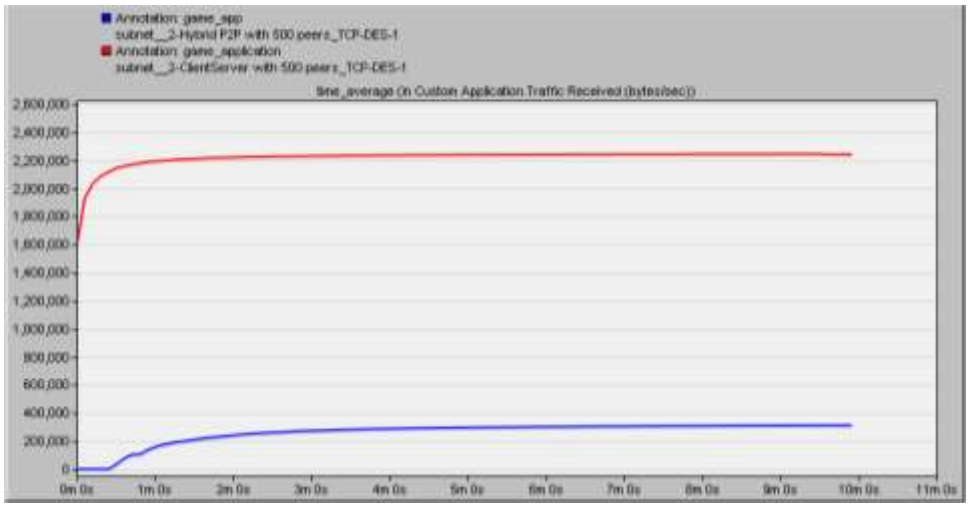


Figure 5. 16: Traffic Received for Hybrid P2P and Client/Server with 500 peers TCP Protocol without Background Traffic

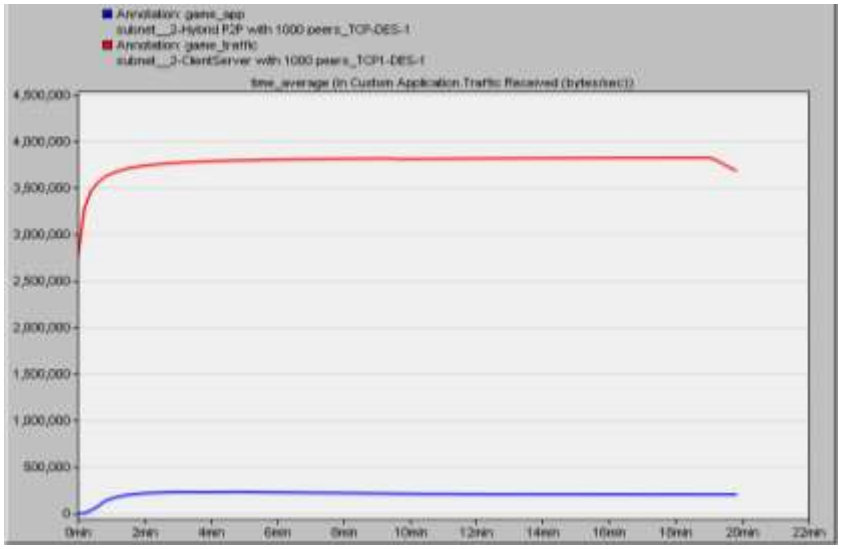


Figure 5. 17: Traffic Received for Hybrid P2P and Client/Server P2P with 1000 peers

TCP Protocol without Background Traffic

5.9.2.4 Background traffic

In order to show the effect of adding the background traffic to the simulation results, we have added some background traffic to all the cable connection in the network model. Table 5.1 shows the background traffic information that was used in our simulation, using uniform X interval 300 seconds/step for both sides. Figures 5.18, 5.19, and 5.20 illustrate the effect on overall delay for adding background traffic to the connection link for MMOGs Hybrid P2P architecture based on custom application and compare the results with MMOGs client-server architecture based on custom application. The results in figures 5.18 and 5.20 show that the background traffic adds few delays to the MMOGs hybrid P2P architecture based on custom application, while it is adding much delay to the MMOGs client-server architecture based on custom application using TCP transport protocol. However, the background traffic does not add any extra delay to both architectures when using UDP transport protocol as shown in figure 5.19.

Table 5. 1: The background traffic information

Seconds	Bits/second
0.0	500.000
300	100.000
600	500.000
900	100.000
1200	500.000

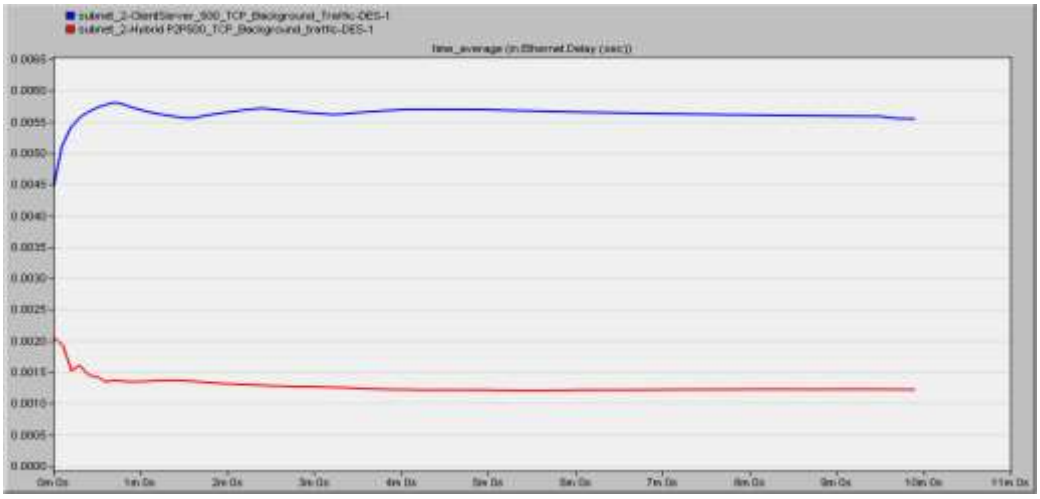


Figure 5. 18: Overall Delay for Hybrid P2P and Client/Server with 125 peers TCP Protocol with Background Traffic

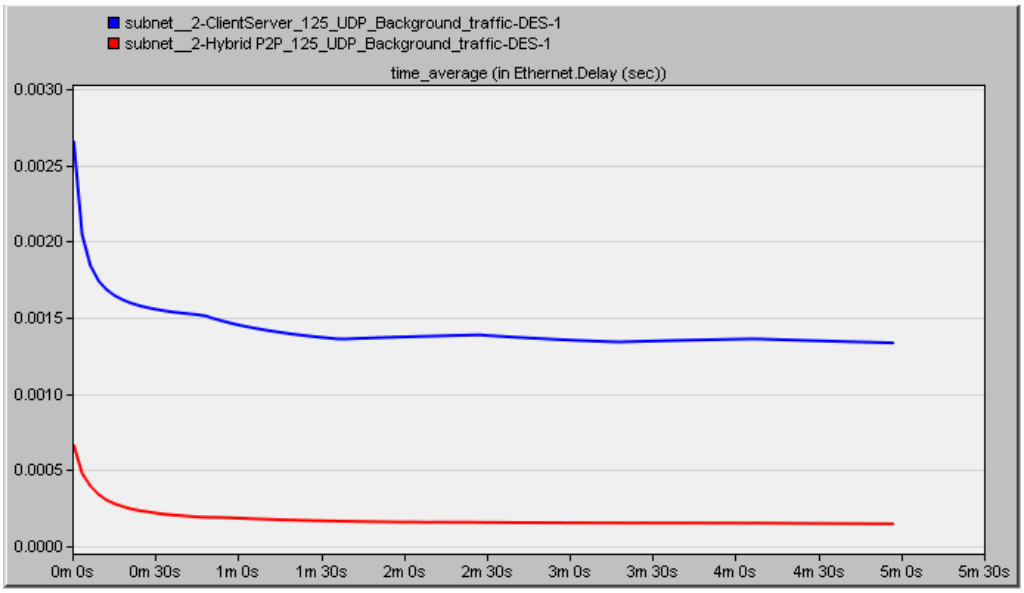


Figure 5. 19: Overall Delay for Hybrid P2P and Client/Server with 125 peers UDP Protocol with Background Traffic

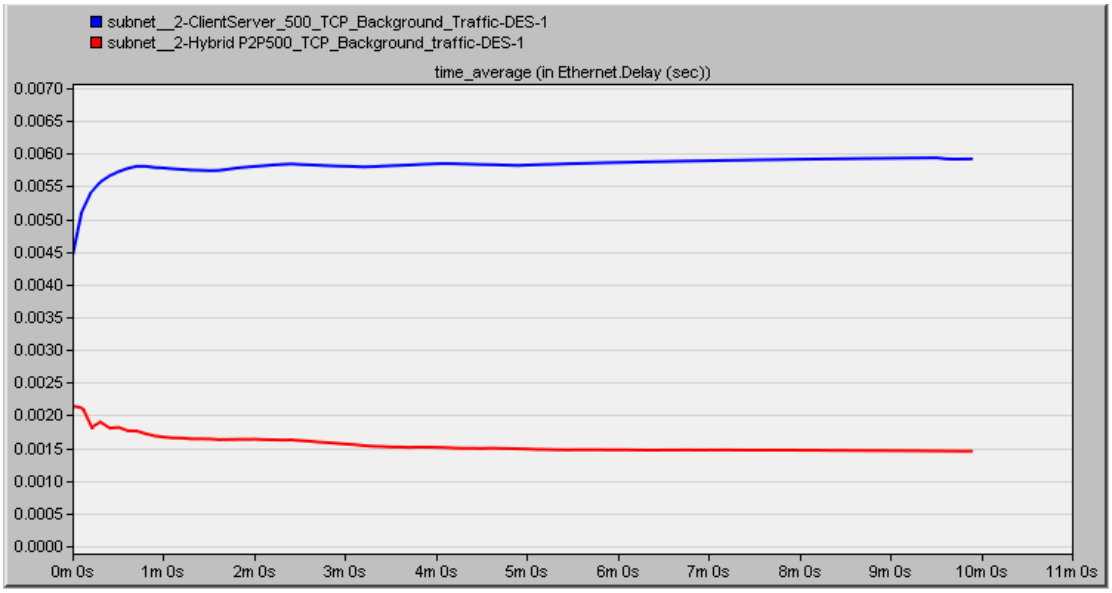


Figure 5. 20: Overall Delay for Hybrid P2P and Client/Server with 500 peers TCP Protocol with Background Traffic

Figures 5.21, 5.22, and 5.23 show the effect of adding background traffic to the connection link for both MMOGs Hybrid P2P and client-server architectures based on custom application. The figures below illustrate that the background traffic does not add any extra traffic received for the three scenarios because the background traffic on baseline affect just the overall delay without impacting the traffic received for the network.

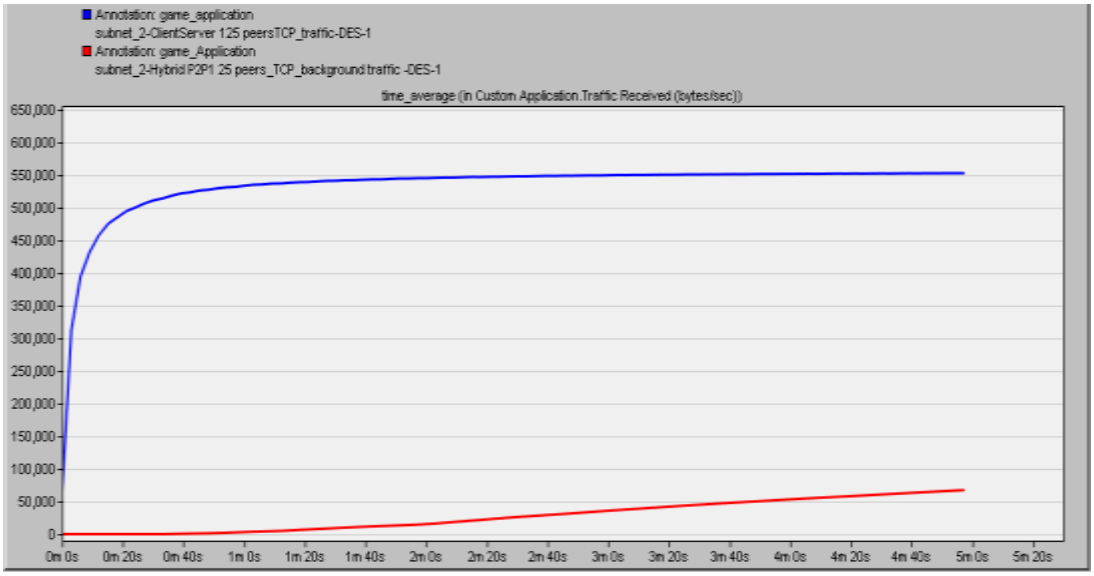


Figure 5. 21: Traffic Received for Client/Server and Hybrid P2P with 125 peers TCP Protocol with Background Traffic

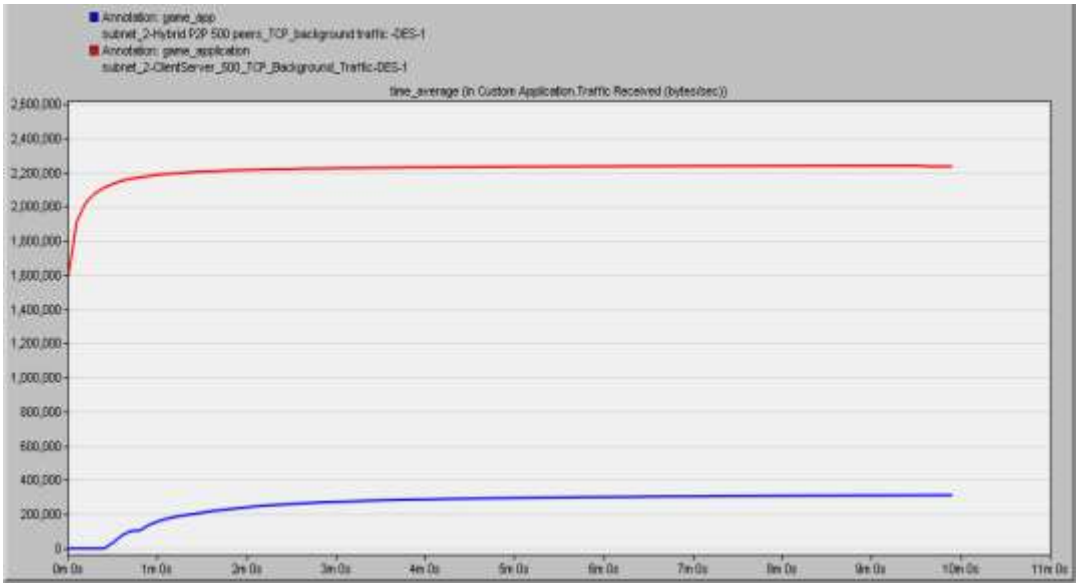


Figure 5. 22: Traffic Received for Client/Server and Hybrid P2P with 125 peers UDP Protocol with Background Traffic

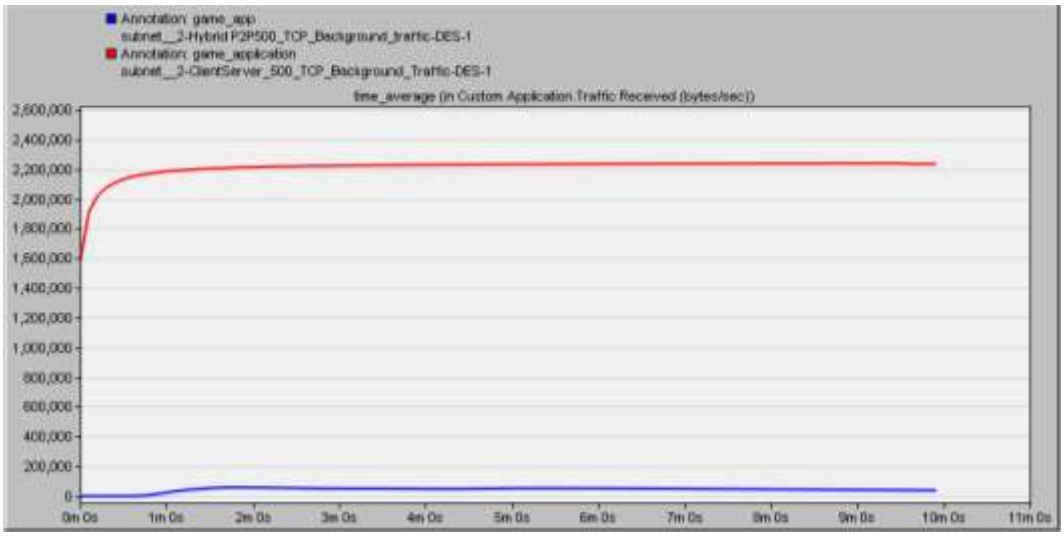


Figure 5. 23: Traffic Received for Client/Server and Hybrid P2P with 500 peers TCP Protocol with Background Traffic

5.10 Discussion

We have designed and simulated a novel MMOGs based on hybrid P2P system and compared the simulation results with the traditional MMOGs based on client-server system. The proposed system provides a good level of scalability, responsiveness, consistency, good level of performance, low latency, an easy way to control and manage the joining and leaving of players, and mechanism for distributing the players among the regions in the game world.

In addition, our system provides a low delay and traffic received compare to the traditional system, therefore it is suitable to apply in the real world. Because our MMOGs based on hybrid P2P architecture provides a good mechanism to distribute the players among the regions and reduce the communication messages and game updates by sending all the updates to the super-peer instead of sending them to all the players inside the game world.

5.11 Conclusion

In this chapter, we have designed and simulated our MMOGs hybrid P2P system based on custom application and compared the simulation results with MMOGs client-server system based on custom application. In addition, allowing the players to migrate from one region to another inside the game world space. The simulation results showed that the use of hybrid P2P system based on custom application provided better results architecture for MMOGs compared with the architecture of the client-server. Also we have obtained good results for the use of MMOGs hybrid P2P system based on custom application with players' movement when compared with the MMOGs client-server system based on custom application with players' movement. In addition to illustrate the effect of adding background traffic to the simulation network.

6

SIMULATION OF AREA OF INTEREST MANAGEMENT FOR MASSIVELY MULTIPLAYER ONLINE GAMES USING OPNET MODELER

6.1 Introduction

A massively multiplayer online games (MMOGs) is considered one of the most important online games capable of supporting typically thousands of concurrently playing players in a shared persistent game world. One of the key important features in these games is the massive number of players who play concurrently over the Internet. The huge number of players leads to a more interactive, complex, and attractive game environment. The essential requirement of MMOGs based on hybrid P2P architecture is to maintain a consistent and shared sense of virtual space between significant numbers of players with minimum resources of game server. Area of Interest Management (AoIM) is considered a conventional research topic that was initially addressed by Macedonia et al. in the mid-1990s [78], to improve the scalability. In order to effectively deploy game updates to the required players, Area of interest management (AoIM) [79] is used to guarantee that just requisite messages will be sent to the group of players involved in the game world. Each player inside the game world has his own area of interest that is commonly represented by a zone around the player. In addition, each player has a list of players that the area of interest covers in the game world, and which corresponds to the users which should receive the requisite messages from the said player. In this research, we have proposed a novel area of interest management mechanism and evaluation using OPNET for our MMOGs hybrid P2P architecture [80] to decrease the delay and network traffic of MMOGs, and avert unnecessary communication of players' state changes. The

concept of AoIM originates from two assumptions: the first assumption is a single player does not need to know about what is happening in the game world as long as it does not impact him; in addition, the second assumption is a player's avatar has limited movement speed and sensing ability, and will not be able to sense (e.g. see, hear, be affected, etc…) by events beyond a certain range [26]. In this chapter, we have designed and simulated a novel area of interest management technique for our hybrid P2P architecture that has been described in chapter five, in order to reduce the communication delay and traffic received of MMOGs even further, and avoid unnecessary communication of players' state changes. The next section explains the zoning concept of AoIM.

6.2 The Zoning Concept for Area of Interest Management

The main goal of using area of interest management is to decrease the cost of data communication in a distributed game, thus reducing delay and, the amount of traffic and bandwidth. This cost relies on the underlying communication architecture, as well as the special area of interest scheme in use. In the next section, we discuss the concepts related to the AoIM. One of the most important concepts in interest management is zoning, whereby, the entire game world is divided into smaller portions, called zones or regions. This is due to the difficulty of maintaining a large number of processes with a single server. Each zone can be dealing with one game server instance. Figure 6.1 illustrates the zoning concept. The player in the zone can only interact with the other players in the same zone. However, the AoIM of a player can change as soon as the player migrates to another zone. The workload can be effectively decreased by reducing the responsibility of a server from maintaining the entire game world. Thus, this mechanism offers a good approach for increasing the scalability of a game server infrastructure. Advanced area of interest management schemes can have the possibility to cover more than one zone. This is significant for continuous game worlds. All players in the region`s borders should be able to interact with objects that are only across the zone boundary in a neighbouring region. Game world can be divided into sub-worlds that are connected to each other such as countries in the game world, with possibility for moving between them [16]. This technique is especially useful when AoI constantly resides within a single region. All the zoning is just virtual, it is not visible to the players, and it is done for the purpose of interest management. In our research, the hexagonal technique was used to divide the game world into regions.

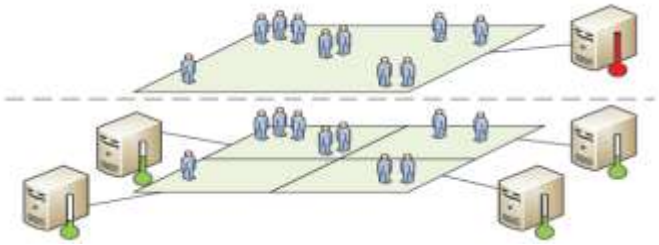


Figure 6. 1: The zoning Concept [94]

One of the simplest approaches is used for the AoIM to be a sub-area of the whole zone. Typically, it is a fixed-radius circle or sometimes a sphere around the player in the zone. Player's AoIM moves accordingly, when the player migrates to another zone. So, interest management must be determined for each player and object in the zone. However, this approach is weak when the players share the same border for two zones. At this situation, the player must choose the suitable zone to connect to it. Figure 6.2 displays an example of the sub-area method.

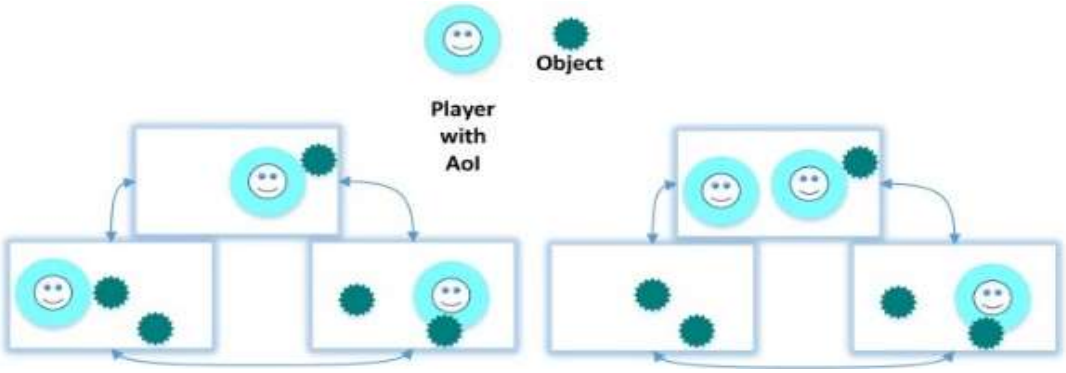


Figure 6. 2: The Sub-area AoI Approach

6.3 Area of Interest Management within P2P

The MMOGs popularity has increased in the last few years and games companies are suffering from scalability problems to accommodate rapid growth in the number of players. Area of interest management is a technique commonly used to determine the smallest amount of information that a peer requires, for the sake of presenting an accurate representation of the game world to players. In other words, area of interest management means communicating information on a need-to-know basis. The aim of using AoIM is to decrease the number of transmitted messages by specifying the potentially interested receivers [15]. This allows communicating the minimum amount of information that a peer needs to interact with other peers in the game world, in order to display an accurate state of the world to the players. This

idea is not only related to P2P MMOGs, but has also been used in client/server architecture for virtual environment [22] and in distributed environment [23]. However, area of interest management becomes an inherent part of the network organisation, when used in conjunction with P2P infrastructures. This method is used with the aim for organising the connections between players in the game world so that they are only contacting with players who possess information pertinent to themselves.

Broadcasting all game state changes to each player in the game world is not an applicable solution to preserve a consistent game state in MMOGs. MMOGs have to use developed area of interest management mechanisms that just send relevant information state changes to each player, in order to overcome successfully the challenge of scale. Providing relevant information to each player in the game world is an efficient method for consistency management.

6.4 The Area of Interest Management Types

There are different approaches and models for implementing the area of interest management for MMOGs. These approaches are described in the next sections.

6.4.1 Publish-Subscribe Area of Interest Management

This model proposes to support MMOGs based on P2P system using coarse-grained AoIM. In this model, the virtual game world is divided into static regions and the recipients of updated messages is limited to just participants who reside within the same area of interest. In this case, AoIM can be done by using a publish/subscribe model, in which publishers are objects that produce events/messages, and subscribers are objects that consume events/messages [79]. An object, for example a player's avatar, can be both a publisher and subscriber. One of the most important responsibilities of Publish-Subscribe AoIM is to determine the area of interest for players in the game world, as well as, to form the area-of-subscription from the union of the intersected regions [79]. The main advantage of publish-subscribe AoIM model is to be simple, easy, and cheaper to compute a player's area of interest management.

6.4.2 Spatial AoIM

One of the simple AoIM is called spatial interest management. It is the mechanism for defining spatial areas or volumes around the player's location in the virtual world [81]. The benefit of a spatial model is that it allows fine-grained interest management in which just

indispensable messages are transmitted among relevant players [26]. However, a significant limitation is that it requires all the objects to interchange positional update information in order to identify when AoIM collisions happen. There are two key abstractions for the spatial areas: aura and nimbus [19]. The "aura" refers to the area that bounds the location of an object in space; however, the "nimbus" means the reciprocal awareness levels between two objects. Thus, each object should start to communicate with other objects that exist within its nimbus, in order to get ready for potential interactions. In its simplest model both the aura and nimbus can be represented by fixed-size circles around the object [19]. The essential advantage of a spatial model is that it allows fine-grained Interest Management in which only requisite messages are transmitted between relevant peers [26]. However, a significant obstacle is that it requires all objects to exchange current update information in order to identify when AoIM collisions happen.

6.4.3 Voronoi Based AoIM

One of the most important techniques developed for Area of Interest Management in P2P networks is based on the concept of a Voronoi diagram [82]. A Voronoi diagram is built by dividing the plane into n non-overlapping regions that include exactly one site in each region. Therefore, each region contains all the neighbours' points closest to the region's site than to any other site. The entire plane is divided into arbitrary sizes in a deterministic method [83]. Figure 6.3 shows a Voronoi diagram, which can be used to locate the k-nearest neighbours of a specific site.

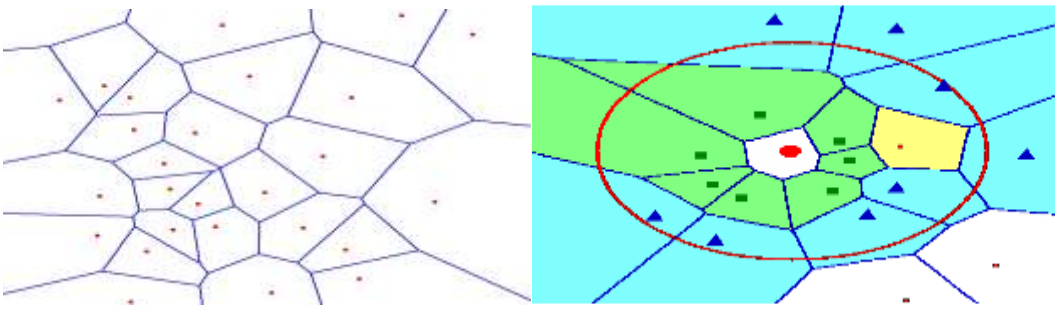


Figure 6. 3: (left) a Voronoi diagram. (Right) Square (□): enclosing neighbours, triangle (Δ): boundary neighbours [97]

Voronoi diagrams have significant advantage in the virtual world, because they allow the distinction to be made between two different kinds of neighbouring peer; Enclosed and Boundary [83]. Enclosed neighbours inhabit a region and share a common edge with a given

node's own region. Nevertheless, boundary neighbours indicate to the regions that overlap with the node's AoIM boundary. When a peer moves from one position to another, position updates are sent to all neighbours recorded in the Voronoi diagram. When the receiver is a boundary neighbour, an overlap check is performed. The receiver reviews if the mover is within its new AoIM, or would enter into any of its Enclosing neighbours' Voronoi regions. The receiver just reports to the mover if a new overlap occurs. This mechanism allows the moving peer to be aware of potentially visible neighbours outside its AoIM. In MMOGs based P2P implementation, each peer is required to build and preserve a Voronoi diagram by itself, based on the spatial coordinates of neighbours. A peer only needs to maintain network connections with its current neighbours, as potential neighbours can be detected with the help of its Boundary neighbours. Position updates are sent to all neighbours recorded in the Voronoi diagram, when a peer moves from one position to another. An overlap-check is performed, when the receiver is a Boundary neighbour. The advantage of using Voronoi diagram is to decrease the communication overhead incurred by a pure spatial model, because every peer only needs to keep network connections and exchange messages with a restricted number of neighbours.

6.4.4 Geographic AoIM

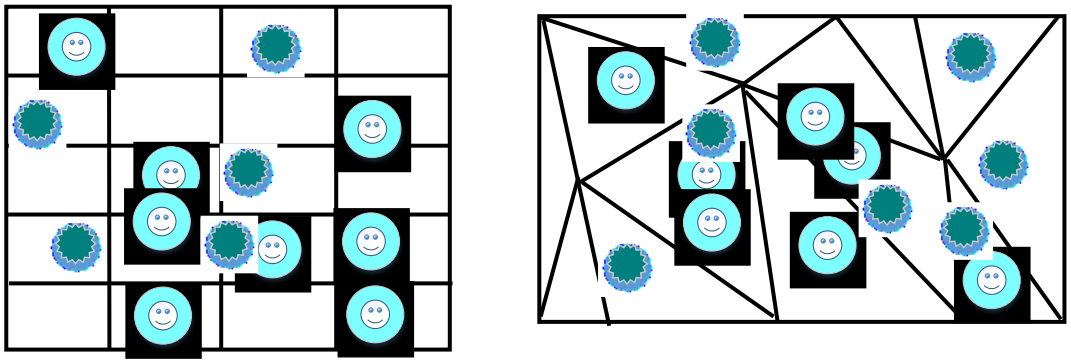
Geographic AoIM refers to the segmentation of the virtual world. It is commonly used to distribute players in MMOGs across several servers. This is a much more fine grained approach in the P2P networks, because massive reduction in computational and networking resource is obtainable on a single peer. In general, there are two types of segmentations: Uniform and Non-Uniform. Uniform methods are used to sub-divide the virtual world into same sized areas; whilst, non-uniform methods are used to divide the virtual world into varying sizes. There are four types of Geographic AoIM and these are described in the following subsections.

6.4.5 Tile-Based Geographic AoIM

The tile-based and commonly so called cell-based method is used to involve a static uniform division of the game world into same size squares, and usually used in MMOGs based on P2P [81], [70]. There are many ways of building the area of interest by using the tile-based technique. The key approach is that players who are within the same tile are interested in each other as shown in figure 6.4(A).

6.4.6 Triangular Geographic AoIM

In this method, the virtual world is divided into triangles. Triangular approaches can be both uniform and non-uniform segmentation. A commonly used method is to use Delaunay triangulation [26]. This technique can help to reduce the AoI and this leads to minimizing the number of objects that have to be considered for interest management, and thus to fewer object replicas to be preserved by the players, as well as fewer update messages that need to be sent [16]. The Delaunay triangulation AoIM mechanism has the property of maximizing the minimum angle in every triangle in the triangulation in order to avoid thin triangles. Also, the authors put a limitation on the maximum triangle area output by the triangulation as shown in figure 6.4(B).



(A) Grid (Tile) **(B) Triangulation (Delaunay)**

Figure 6. 4: Geographic AoIM (Tile, Delaunay triangulation) [37]

6.4.7 Hexagonal Geographic AoIM

The hexagonal AoIM technique is used to divide the virtual world into a uniform hexagonal pattern [84] [85]. This constrains the number of neighbouring regions and provides an optimal approximation of a radial AoIM surrounding a player. In our research, we have used the hexagonal geographic technique to represent each region in the game world space as shown in figure 6.5. The reason for choosing this technique is to have a uniform approach and uniform adjacency. This means the players will typically migrate to a close region in the game world.

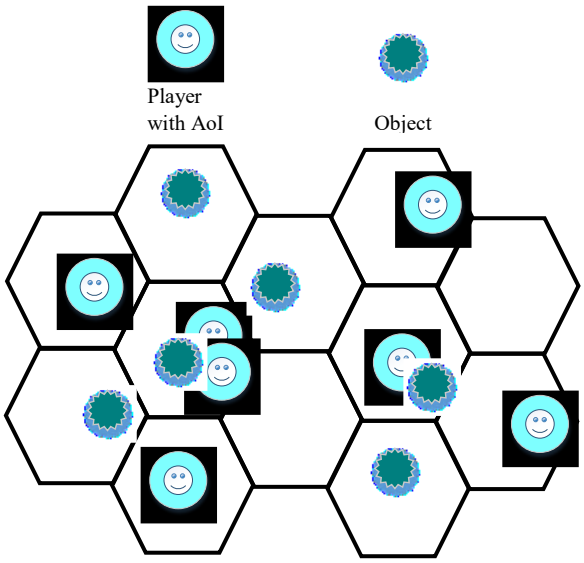


Figure 6. 5: Hexagonal AoIM

6.4.8 Hierarchical AoIM

This method is used to divide the virtual world into hierarchical divisions in order to introduce dynamism and non-uniformity into geographic AoIM [73]. This approach usually applied using a tree-structure, also a method used within the game engine to administer three-dimensional scenes by using techniques for instance Binary Space Partitioning (BSP) [86], QuadTrees and Octrees [87]. Figure 6.6 gives a good illustration of this approach.

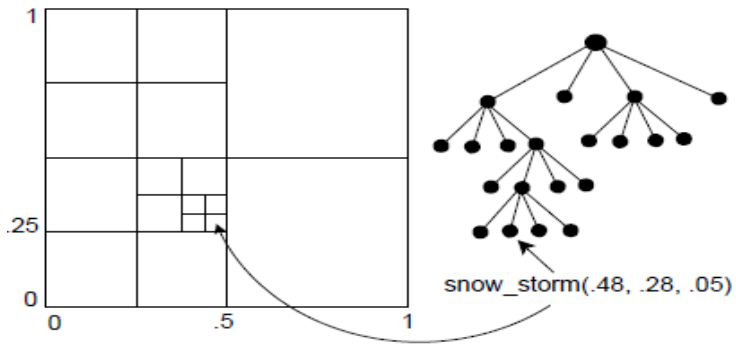


Figure 6. 6: Hierarchical AoIM using Quad Tree [87]

6.5 Static Area of Interest Management for MMOGs

The convenient management of AoIs in MMOGs is a challenging work. With the considerable increase in player numbers in MMOGs, Area of interest management has become one of the most significant features in the MMOGs field. Several researches have been proposed to address the area of interest management. However, to the best knowledge of the

authors, there is no research which has proposed to deal with the area of interest management of MMOGs based on hybrid P2P architecture and combined with dynamic load balancing which will be introduced in the next chapter. We have proposed to design a static area of interest management for MMOGs based on hybrid P2P system. The main idea of the AoIM proposed system is to create a list of receiver players inside each region in the game world space. The AoIM static list consists of a number of players in the same vicinity. The game world space is divided into several regions. Each region in the game world is managed and maintained by using both region's super-peer and clone-super-peer. The game server is only responsible for the players' registration of game and manages to assign the super-peer for each region. Region super-peer is responsible for creating the AoIM lists and sending the game update only to the players in the AoIM static list instead of sending to all the players in the region. In order to exploit further the ability of AoIM to improve scalability, we propose to refine the model further and use AoIM inside each individual region the discriminate the messages communication from one to only those nodes who are in its list within a region.

6.6 Simulation Design for Static AoIM

Using OPNET Modeler 18.0 custom application [42] to design and simulate the proposed system by creating a static list of players for MMOGs based on hybrid P2P architecture and compare the results with the traditional MMOGs client-server system. In hybrid P2P architecture, we have created lists of interest players by using both region super-peer and clone-super-peer. The game updates for the player are done by sending the update to all the players in the list of interest, not to the all players in the region. However, the lists of interest in MMOGs based on client-server architecture have been created by using the game server. All the game updates and the players' communication are done by the server. When the players exist in two different lists of interest, they will receive game updates twice. When new players join the game, the super-peer in MMOGs based on hybrid P2P system or the server in MMOGs based on client-server system will add them to the suitable list of interest. However, when the player leave the game world, they will be removed from the list of interest. The criteria used for selecting the players for list is the player's location in the game world. The communication between the players inside the region will be the responsibility of the region's super peer in MMOGs hybrid P2P architecture. Every game update must be sent to the region super peer so they are able to send them to the players inside the region. The main advantage of using static area of interest management are to send the game updates just to the players in the list instead of sending them to all players in the region or in the game world, and reducing

drastically the amount of communication required to share the game updates. The network topology consists of two main scenarios for each architecture which are explained in the previous work [80]. The experiments will be carried out with scenarios using 125, 500 and 1,000 peers consecutively, and evaluating architectures based on client/server, client/server with AoIM, Hybrid P2P and Hybrid P2P with AoIM. We have used TCP transport protocol in all the scenarios. The number of players per list of interest is around 20-30 players. Figures 6.7 and 6.8 show the network design for static AoIM for both MMOGs hybrid P2P and MMOGs client-server respectively.

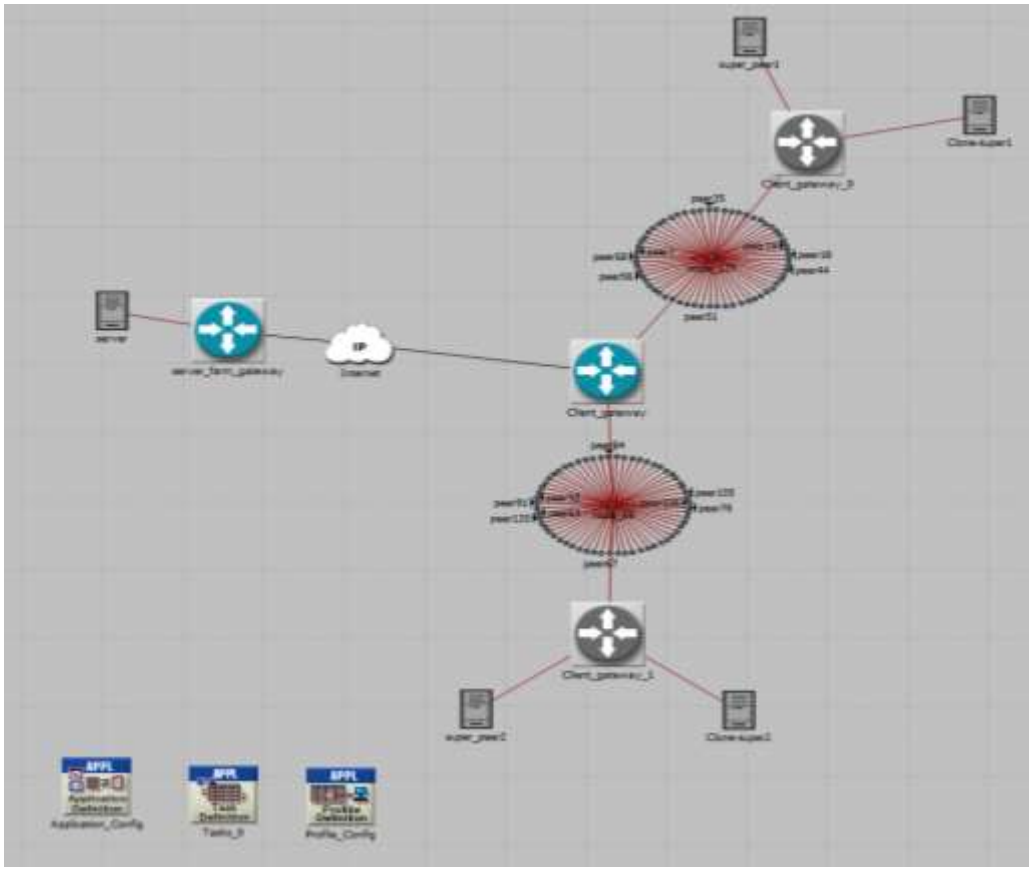


Figure 6. 7: MMOG hybrid P2P architecture with static AoIM

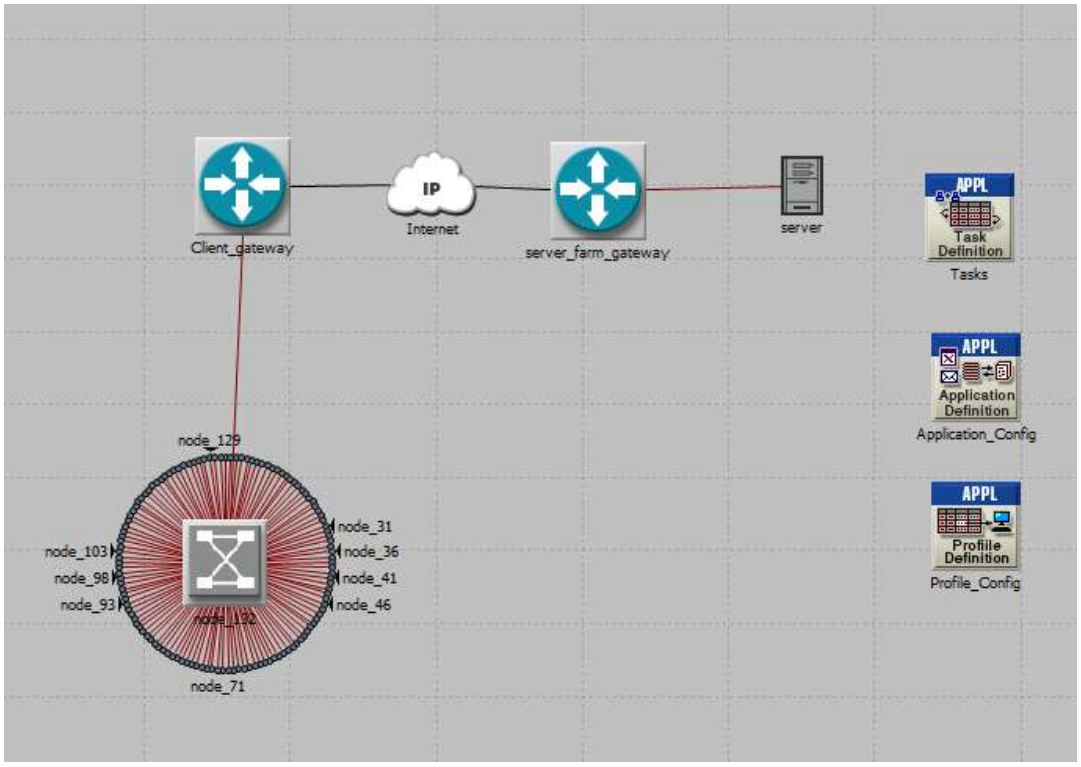


Figure 6. 8: MMOG client-server architecture with static AoIM

6.7 Simulation Results for Static AoIM

With the scenarios that have been introduced previously, we have got the following simulation results

6.7.1 Overall delay

Figures 6.9, 6.10 and 6.11 show the overall delay results for the MMOGs based on hybrid P2P architecture with and without AoIM and compare the results with MMOGs based on client-server architecture also with and without static AoIM. The figures below illustrate a small variation of delay when using hybrid P2P system with AoIM compared with the system without using AoIM. However, the variation of delay is more significant when using client-server system. Nevertheless, the overall delay for MMOGs based on hybrid P2Psystem with AoIM is very small when compared with the MMOGs based on client-server architecture.

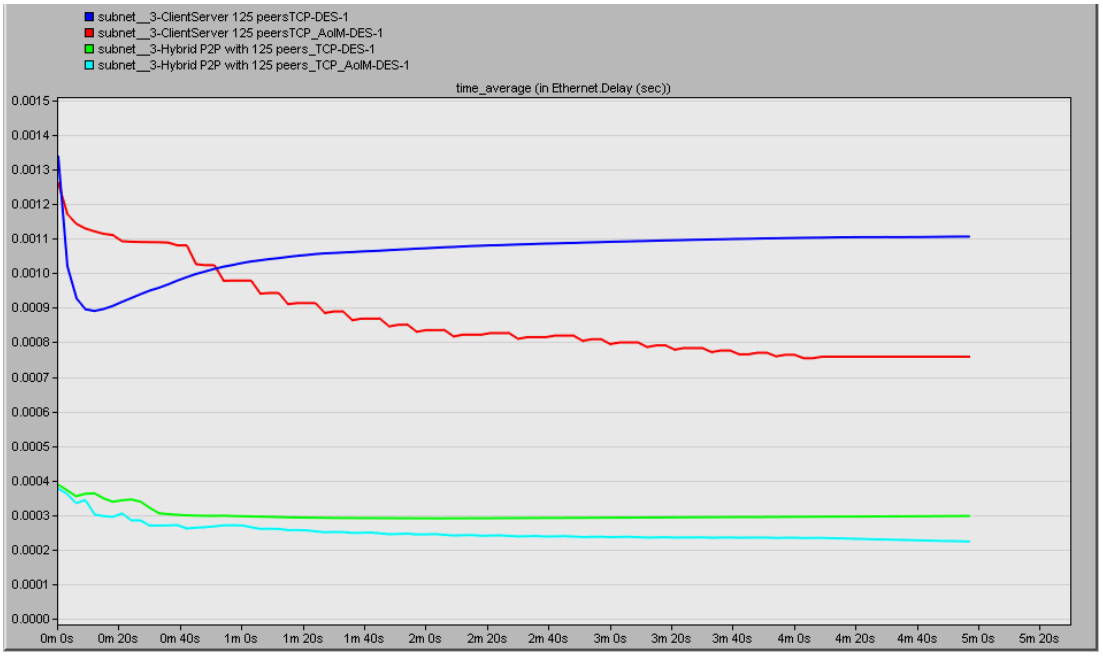


Figure 6. 9: Overall Delay for Client/Server and Hybrid P2P with 125 peers with Static AoIM

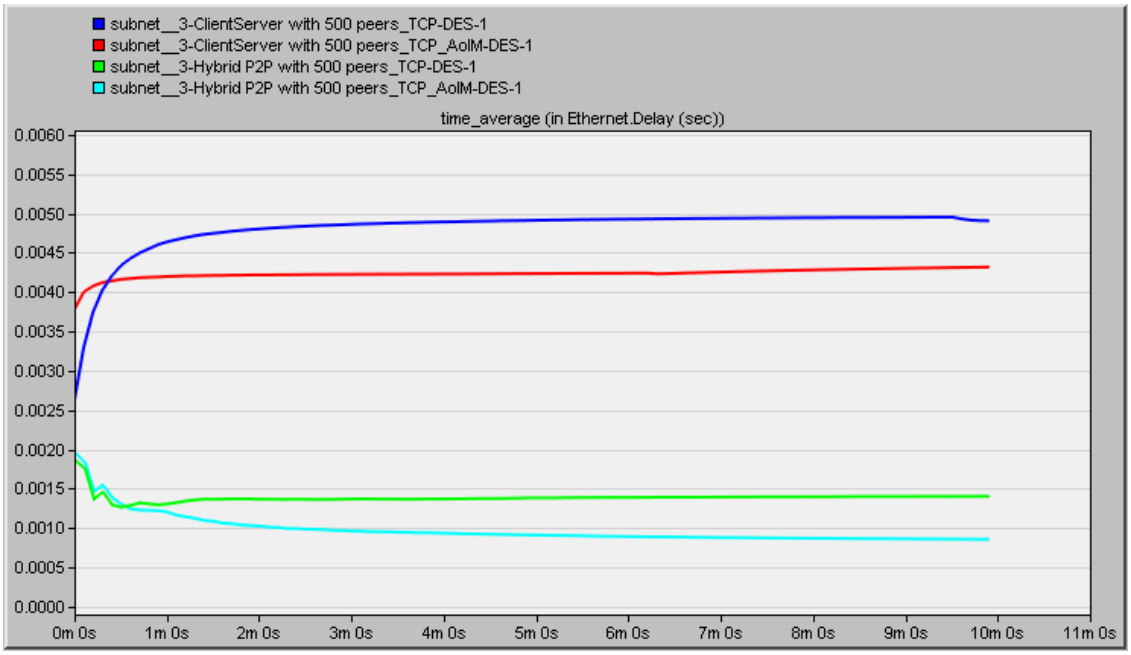


Figure 6. 10: Overall Delay for Client/Server and Hybrid P2P with 500 peers with Static AoIM

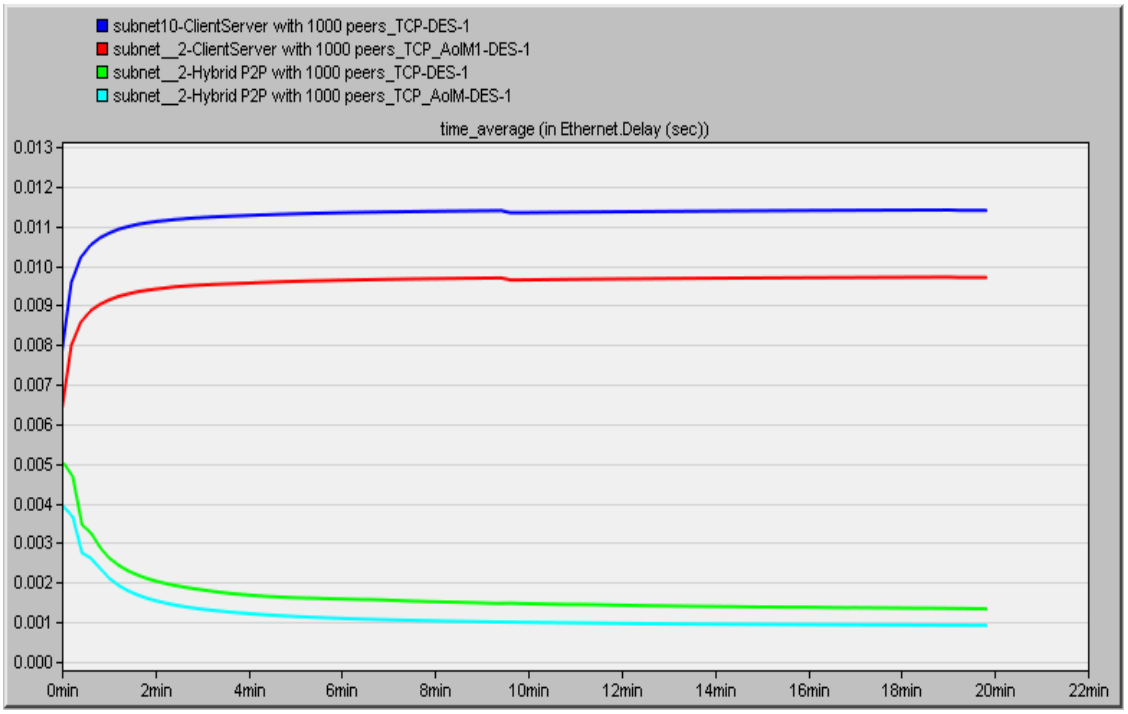


Figure 6. 11: Overall Delay for Client/Server and Hybrid P2P with 1000 peers with Static AoIM

6.7.2 Traffic Received

Figures 6.12, 6.13 and 6.14 illustrate the simulation results of traffic received for the MMOGs based on hybrid P2P with AoIM and compare the results with MMOGs based on hybrid P2P without AoIM. As well as, the same for MMOGs based on client server architecture. The figures below illustrate a small variation of traffic received when using hybrid P2P system with AoIM compared with the system without using AoIM. However, the variation of traffic received is significant when using client-server system. Nevertheless, the traffic received for MMOGs based on hybrid P2Psystem is very small compared with the MMOGs based on client-server architecture.

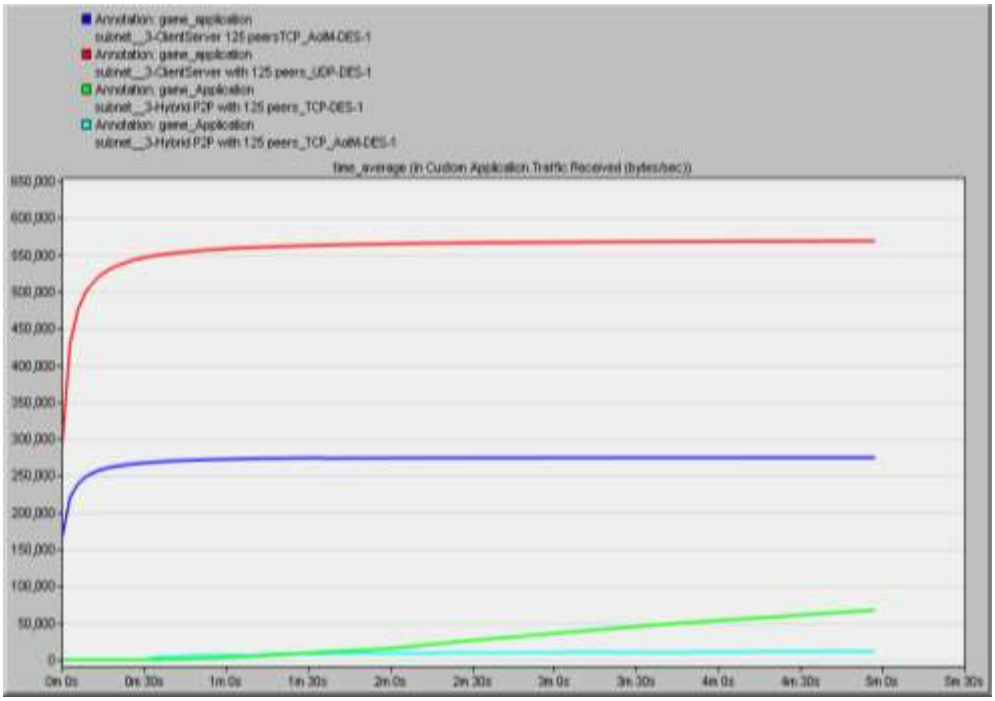


Figure 6. 12: Traffic Received for Client/Server and Hybrid P2P with 125 peers with Static AoIM

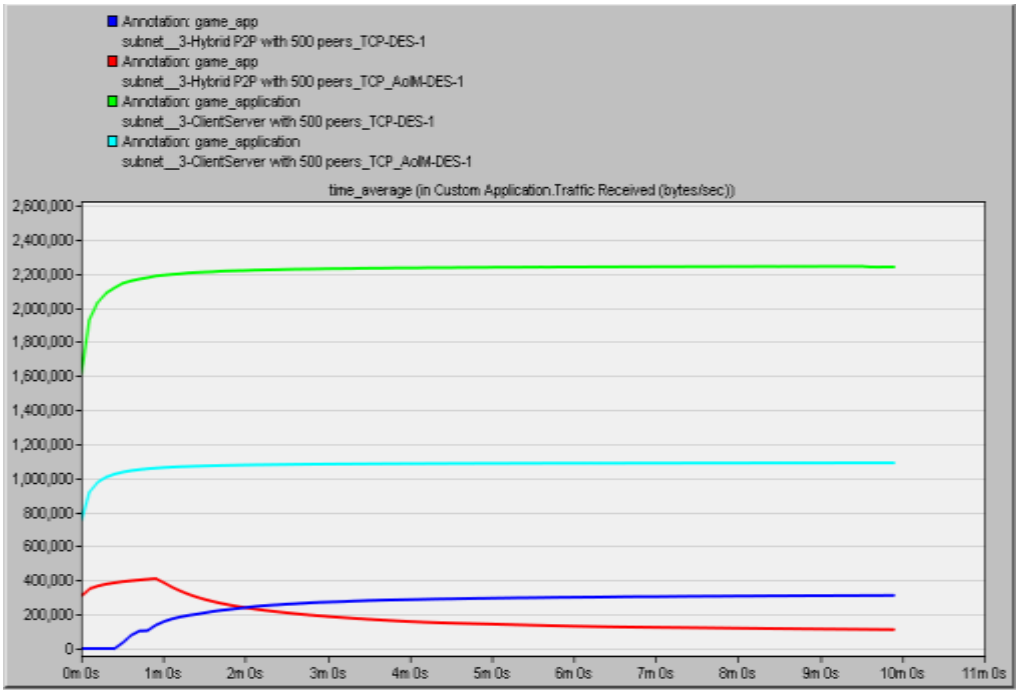


Figure 6. 13: Traffic Received for Client/Server and Hybrid P2P with 500 peers with Static AoIM

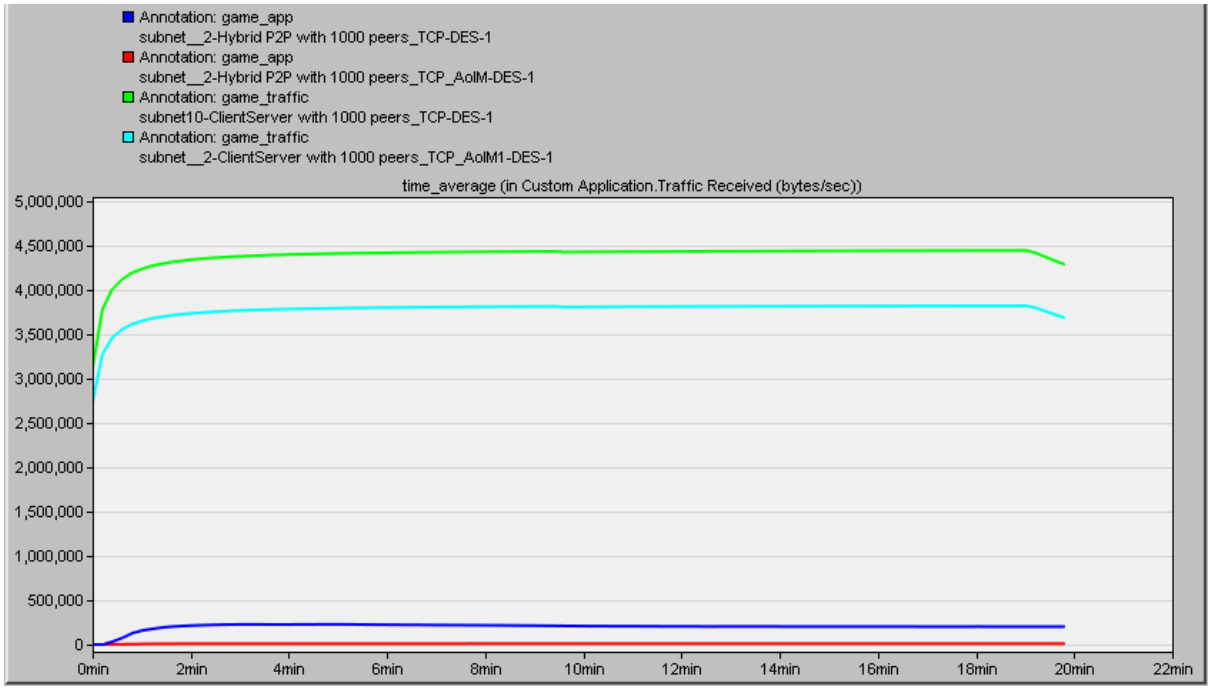


Figure 6. 14: Traffic Received for Client/Server and Hybrid P2P with 1000 peers with Static AoIM

6.8 Dynamic Area of Interest Management

6.8.1 Wireless Network for MMOGs

In OPNET Modeler, there are two kinds of communication wireless nodes: Mobile Nodes and Satellite Nodes. The two kinds of wireless nodes are similar in fixed communication nodes, however they can change the position during the simulation. In our simulation, we have used the Mobile nodes in order to use them to change the position of the node, as well as the migration of players form one region to another in the game world space. Mobile communication node has the possibility to change positions during the simulation by using a pre-defined trajectory. The node's position is automatically updated at suitable times to follow the specified trajectory during the simulation. We have used the wireless network devices instead of wired for several reasons. The first reason is the ability to move and change the position during the simulation. The second reason is most of the devices in this time are wireless such as mobile phone, iPad, tablet, laptop. The third reason is the wireless network for MMOGs hybrid P2P system produces less delay and traffic received compared to the wired network for MMOGs client-server system as shown in figures 6.15 and 6.16.

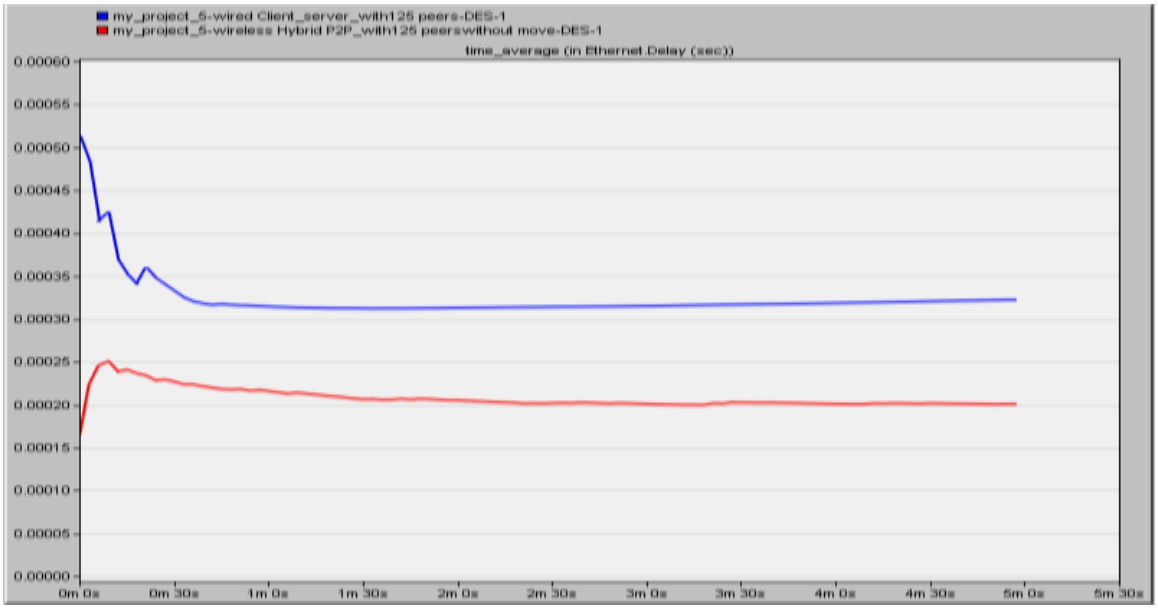


Figure 6. 15:Overall delay for Wireless MMOGs Hybrid P2P Compared to Wired MMOGs Client-Server

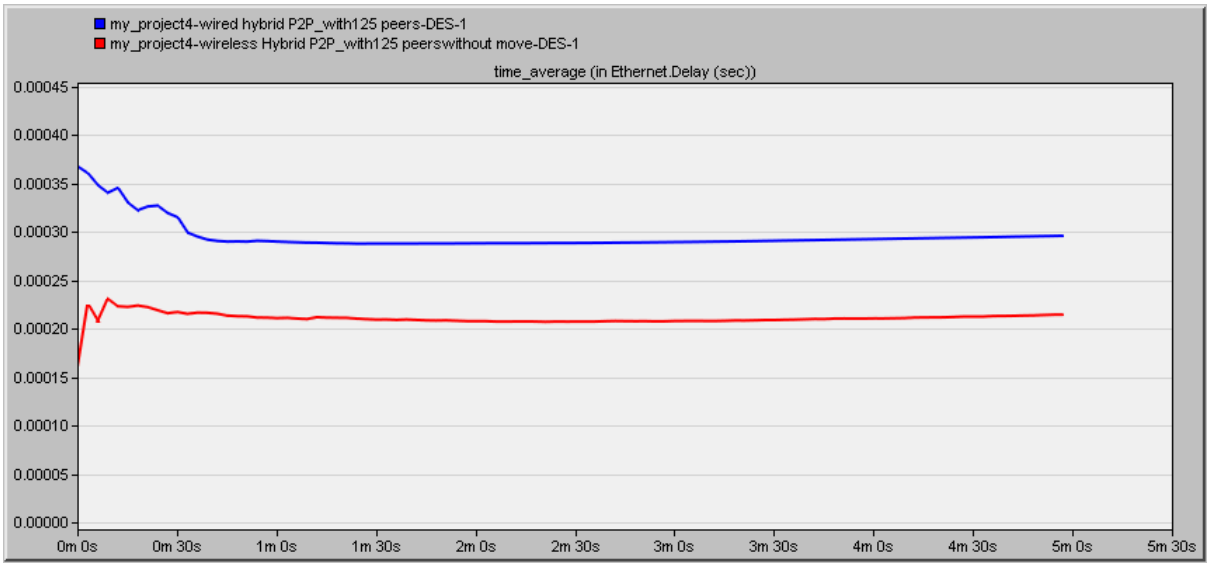


Figure 6. 16: Overall delay for Wireless MMOGs Hybrid P2P Compared to Wired MMOGs Hybrid P2P

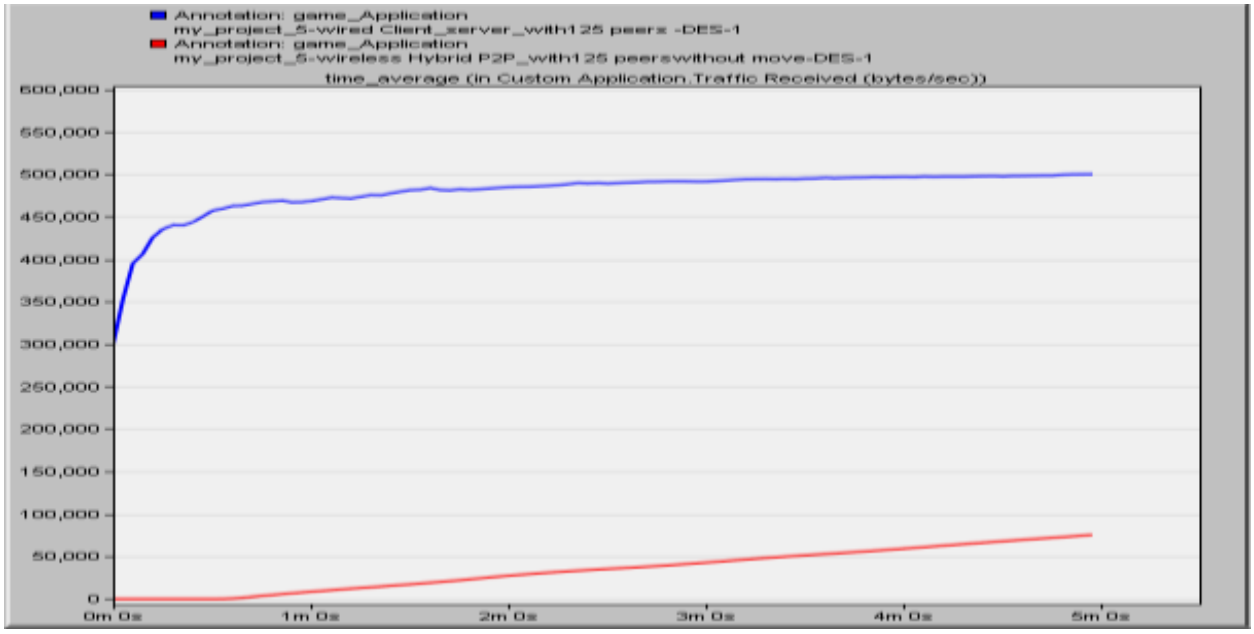


Figure 6. 17: Traffic Received for Wireless MMOGs Hybrid P2P Compared to Wired MMOGs Client-Server

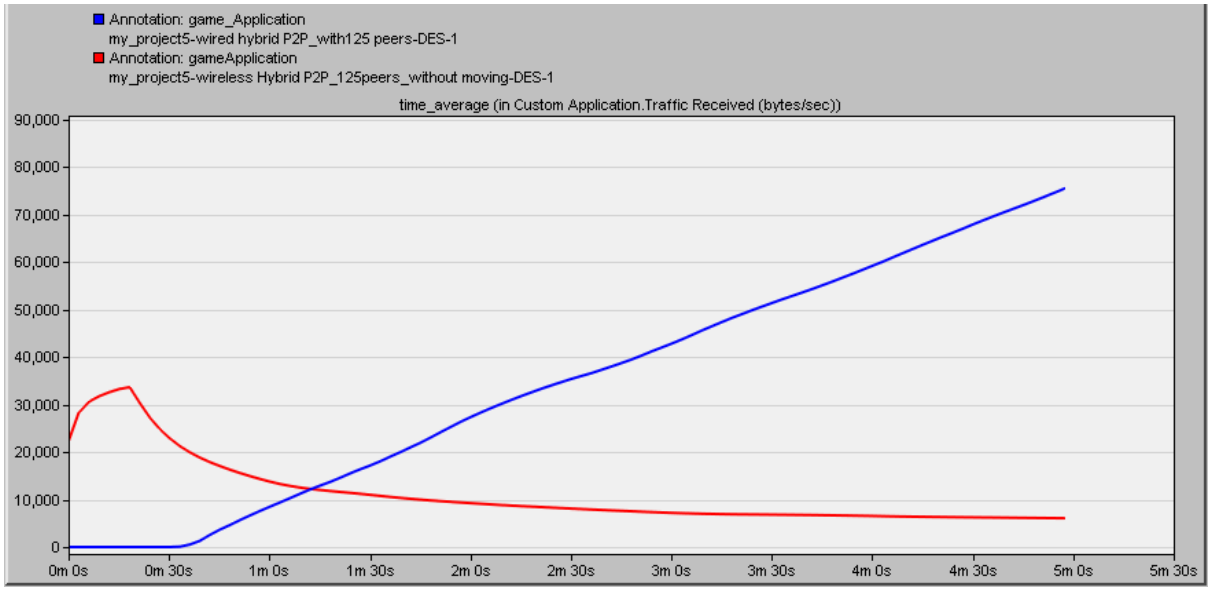


Figure 6. 18: Traffic Received for Wireless MMOGs Hybrid P2P Compared to Wired MMOGs Hybrid P2P

6.8.2 Node Movement

Wireless network communication is considered a broadcast technology and depends on dynamically changing attributes, especially the distance in order to evaluate the potential connectivity between a transmitter and receiver devices. OPNET Modeler uses the distance

between network nodes to calculate link influences such as delay, and traffic received power. One of the important characteristics of the performance of a system related to wireless node is the possibility for moving from one position to another inside the game world space. Thus, the good movement models of nodes are decisive for many simulations of communication networks. The Wireless network provides a mobility function that can allow for nodes to change location depending on predefined trajectories such as organized orbits or randomly selected paths. The locations of wireless node are automatically updated during simulation according to the trajectory for each node. Riverbed Modeler has the possibility of creating custom trajectory movement of wireless nodes. Riverbed Modeler also allows the users to model movement of wireless nodes by manipulating position attributes. Each network node must start its movement within the subnet range of its parent subnet. After the start of simulation, the network node can exceed the boundaries of the parent subnet [77]. Figures 6.19 and 6.20 show the screen shot of MMOGs hybrid P2P architecture with peers' movement from one region to another. These figures illustrate the migration of players from one region to another to player there. However, figure 6.21 and 6.22 show the overall delay and traffic received respectively for MMOGs hybrid P2P of 125 peers with movement compare to MMOGs hybrid P2P of 125 peers without movement. The figures below show that the MMOGs based on hybrid P2P architecture with players movement provide more delay and network traffic received. Because the frequently change of the player's position in the game world space, as well as the players prefer to move to special area to play there as shown in figure 6.23.

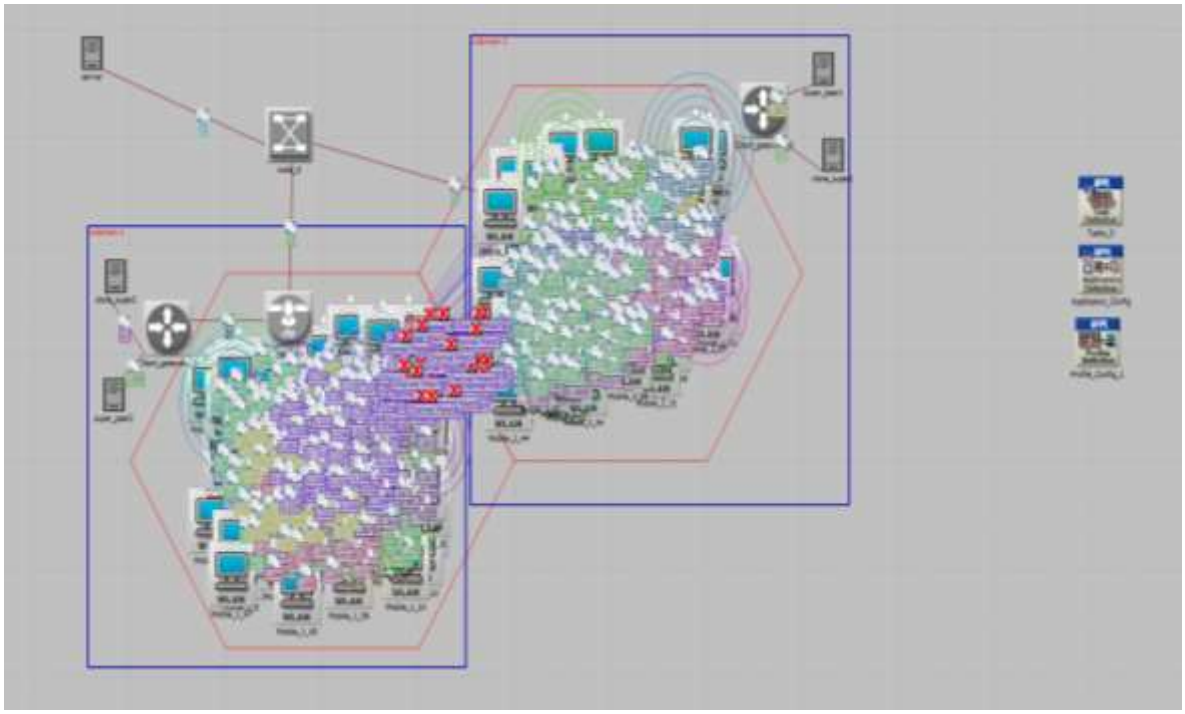


Figure 6. 19: Players Movement for MMOGs Hybrid P2P of 125 Peers

Figure 6. 20: Players Movement for MMOGs Hybrid P2P of 125 Peers

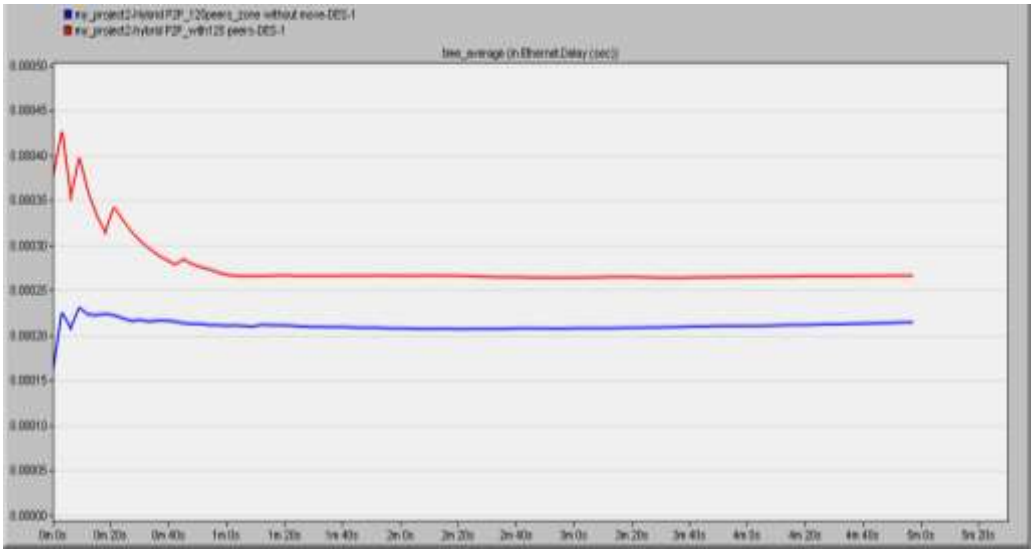


Figure 6. 21: Overall delay for MMOGs hybrid P2P of 125 peers with movement

Compared to MMOGs hybrid P2P without movement

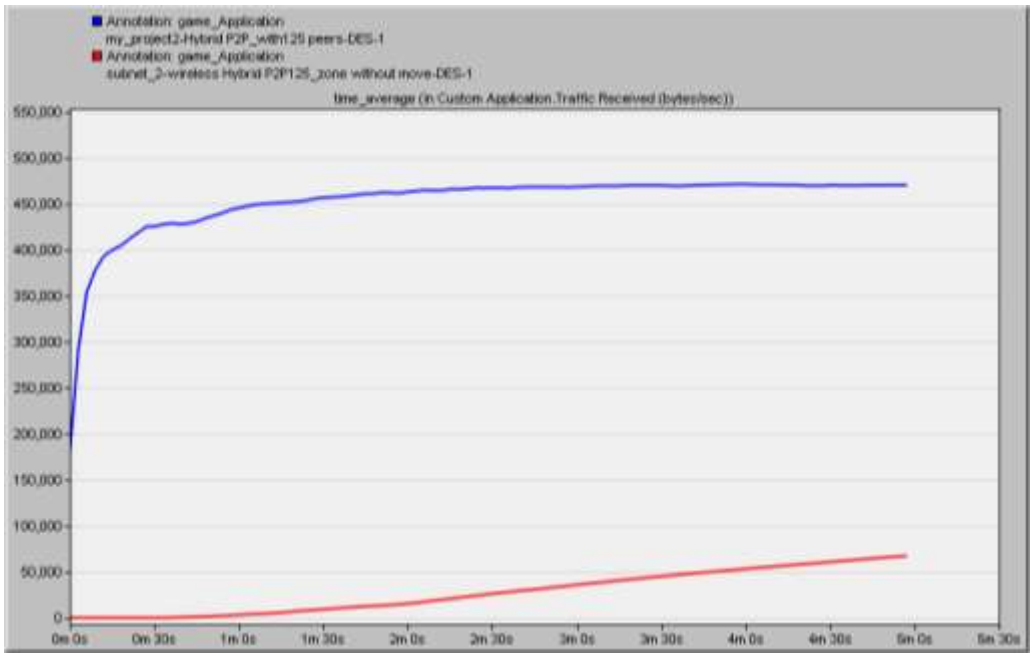


Figure 6. 22: Traffic Received for MMOGs hybrid P2P of 125 peers with movement

Compared to MMOGs hybrid P2P without movement

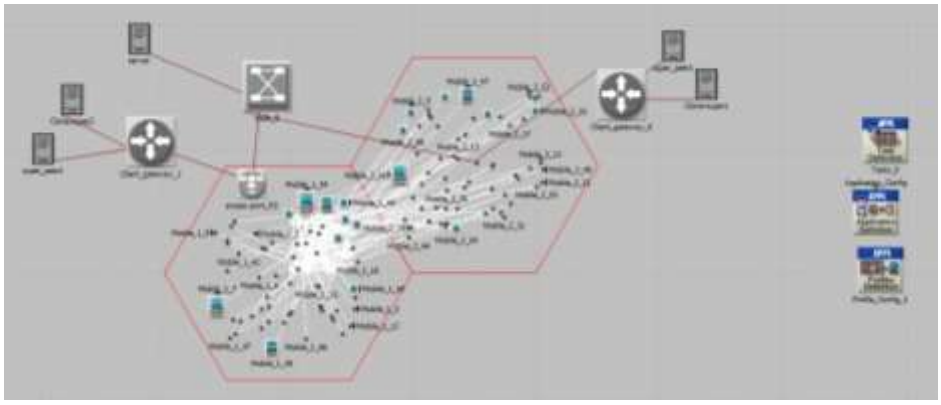


Figure 6. 23: Screen Shot for MMOGs hybrid P2P of 125 Peers' Movement

6.9 Dynamic AoIM for MMOGs

The traditional MMOGs have used the client/server system to maintain and develop online games. This architecture is really expensive compared to the MMOGs based on P2P architecture, due to the huge number of servers that are used to support the virtual space. Also the maintenance and the development of such server-clusters are extremely expensive costing millions of pounds. The use of MMOGs based on hybrid P2P architecture [88] is considered the alternative solution for both players and game companies. Each player in the game world must preserve a copy of the relevant game states on his/her computer, in order to guarantee a consistent game space for the players. The game world space is influenced when a player creates or executes an action or even updates a game, thus, the game state of all other players is affected by that event or action. Therefore, the amount of information required for communication among players depends on the number of players in the interested area. AoIM is considered one of the most efficient and straightforward mechanisms to determine the group of a player's relevant entities. In this research, we have proposed a dynamic area of interest management for MMOGs based on P2P architecture. The game world is divided into several regions. Hexagonal AoI technique is used to divide the game world into regions. These regions have been built before the simulation. The hexagonal technique has some features over the others, for example preserving a regular format in order to enable regions to stick together without having any gap between them inside the game world, which could happen when we choose circles as region shapes. The reason for choosing this technique is to have a uniform approach and uniform adjacency. This means the players will typically migrate to a close region in the game world. After a player has joined the game, the player starts to create a dynamic list of possible receivers that affects the game state of the player (neighbours surrounding the player position). These lists of receiver are updated when the new game update is received from

the super-peer of the region. These lists will be sent to the super-peer and clone-super-peer of the region. If the player moves inside the region or migrates to another region, dynamically, it will begin to create a new list of receivers, and also send the new list to the region's super-peer and clone-super-peer. Region's super-peer sends all the game updates to the players on the list instead of sending it to all the players in the region. If the player left the game, it will be removed from the list of receivers, and it will not be visible to other players in the region. In order to insure a high level of scalability, the game server is responsible for choosing the right super-peer for each region depending on the resources of super-peer such as memory, CPU, disc space, and bandwidth. The players joining and leaving the game are controlled and managed primarily by the region's super-peer. When the player leaves the game, the player will inform the super-peer of the region, then the list is dynamically deleted from both player and super-peer. If the super-peer has formally left the game, it will inform both the game server and the clone-super-peer to take the responsibility and be the super-peer of region. However, if the has super-peer suddenly leaves the game, the player will directly send all the messages to the clone-super-peer according to the selection weight of the application attribute. In the case of the clone-super-peer left the game, the super-peer of the region is responsible for choosing the suitable player to be the clone-super-peer according to the resources of the player in the game world. In all cases, it must inform the game server about the movement, migration, and leaving. In our architecture, we have the possibility of both super-peer and clone-super-peer to migrate to another region as a normal player.

6.10 Simulation Design for Dynamic AoIM

We use OPNET Modeler 18.0 custom application [43] to execute and simulate the proposed system and compare the results with the traditional architecture client-server architecture. In MMOGs hybrid P2P architecture, each player is responsible for creating a list of interest players surrounding the player position in the game world space. The player is also responsible for sending this list to the region's super-peer in order to send all the game updates to the players involved. When new players join the game, the player starts to create a list of interest players, however, if the player leaves the game, the super-peer of the region will destroy the list of interest. Figures 6.24, 6.25 and 6.26 show the design of AoIM for MMOGs hybrid P2P architecture with 125, 500, and 1000 players respectively. In the figures below, we have used TDMA network device in order to provide a dynamic connection and create a dynamic list of receiver for each player in the game world space. Also we have used the RX Group configuration to set up the three main criteria used to select the dynamic AoIM list of players

[69]. These criteria are explained in the next section. We illustrate the AOIM technique using nodes connected wirelessly in OPNET Modeler.

- **Receiver Group**

The receiver group criterion is determined by the "rxgroup model" attribute of the network. When a player starts the game, the broadcast nature of network device execute several radio links between the player and a group of receiver channels. Each transmitter player in the game world space maintains its own receiver group of players that are potential candidates for receiving message from that player.

- **Closure**

The closure criterion is determined by the "closure model" attribute of the network device. It is invoked once for each player referenced in the receiver group. These invocations happen directly after the returning from the receiver group criterion, without simulation time elapsing in between. The main objective of this criterion is to specify whether a special receiver player can be influenced by a transmission player. The capability of the transmission player to reach the group of receiver channel is indicated to as closure between the player and the receiver group of players in the game world.

- **Channel Match**

The channel match criterion is determined by the "chanmatch model" attribute of the network device. It is invoked once for each player that satisfies the criteria of the link closure criterion. These invocations happen directly after the returning from the link closure criterion, without simulation time elapsing in between. The key purpose of this criterion is for classifying the transmission player with respect to the receiver channel group.

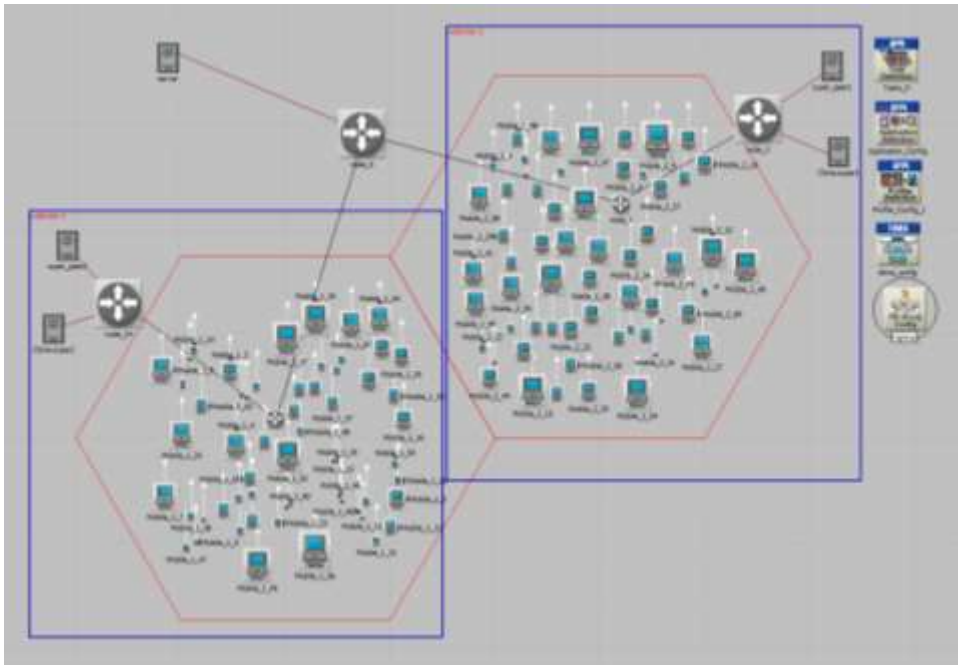


Figure 6. 24: Dynamic AoIM of MMOGs Hybrid P2P System with 125 peers

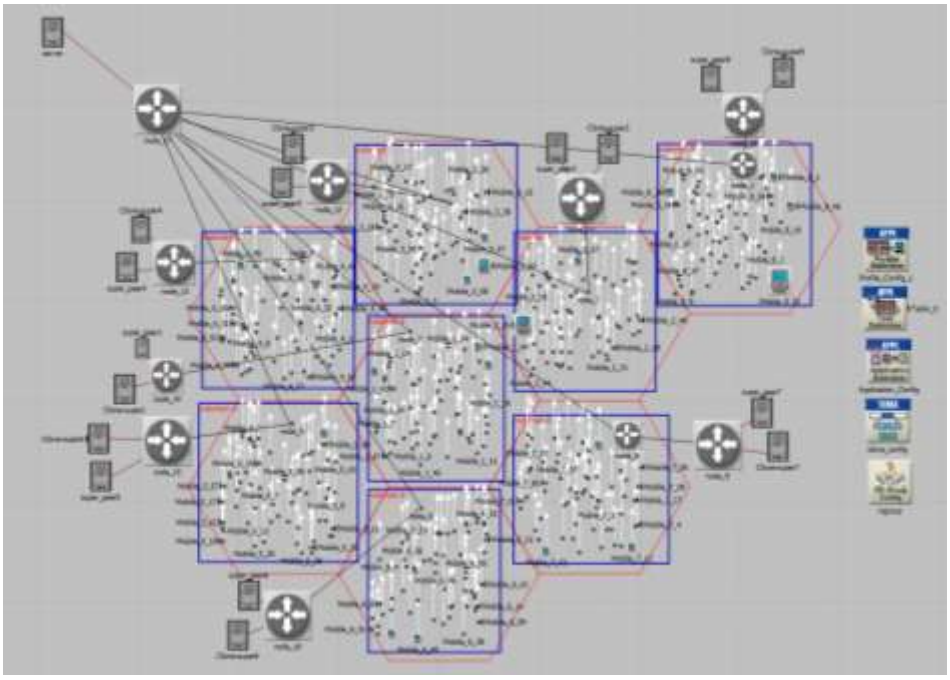


Figure 6. 25: Dynamic AoIM of MMOGs Hybrid P2P System with 500 peers

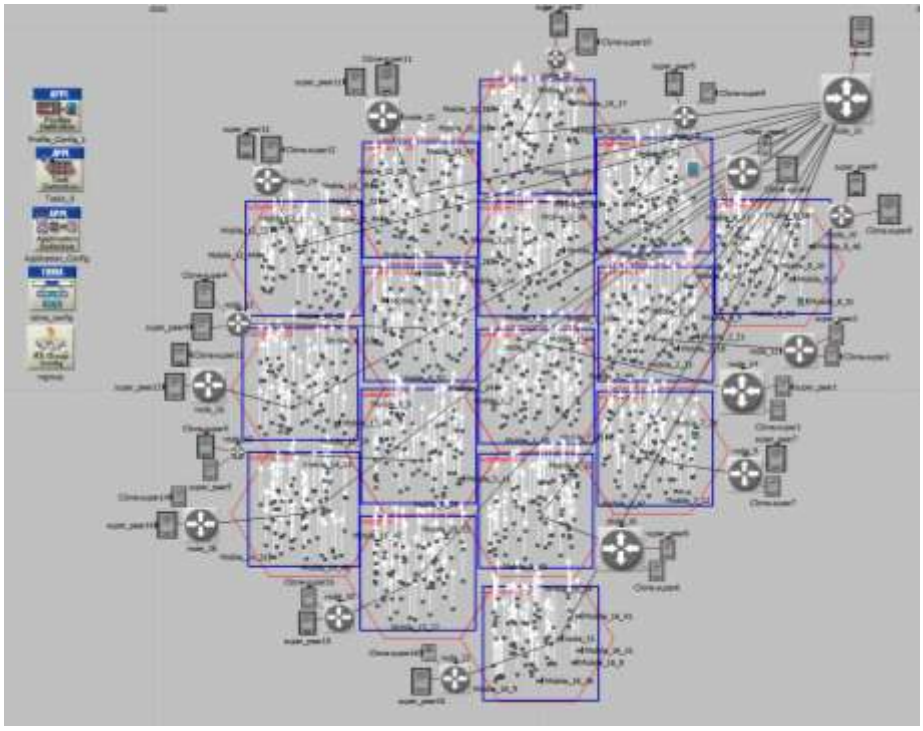


Figure 6. 26: Dynamic AoIM of MMOGs Hybrid P2P System with 1000 peers

6.11 Simulation Results

In this section, we have analysed the network performance under simulation scenarios from the view of three primary parameters.

6.11.1 Simulation time

The simulation time is known on OPNET Modeler as Elapsed time. It is the actual time taken to complete the simulation. This time varies from one scenario to another depending on the different factors such as simulation duration, network topologies (number of nodes and type of application), and transport protocol. Dynamic AoIM for MMOG hybrid P2P system provides lower Elapsed time compared to the MMOG hybrid P2P system without AoIM as shown in figures 6.25 and 6.26.

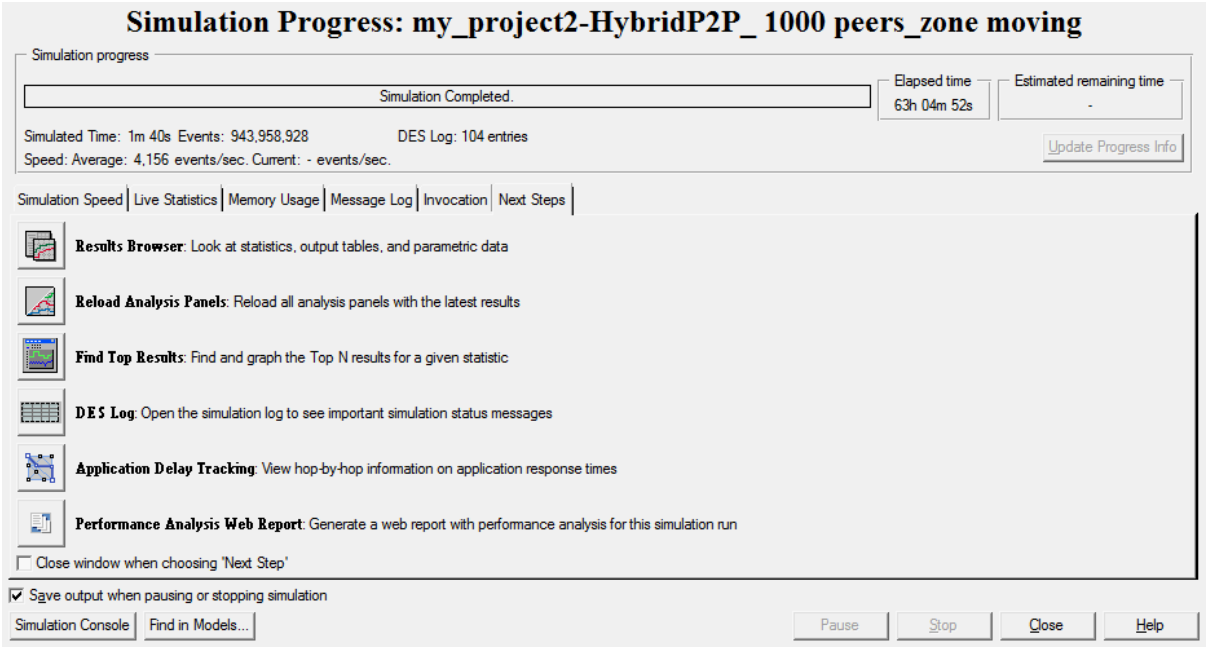


Figure 6. 27: Simulation Time (Elapsed time) for MMOGs Hybrid P2P System with 1000 peers without AoIM

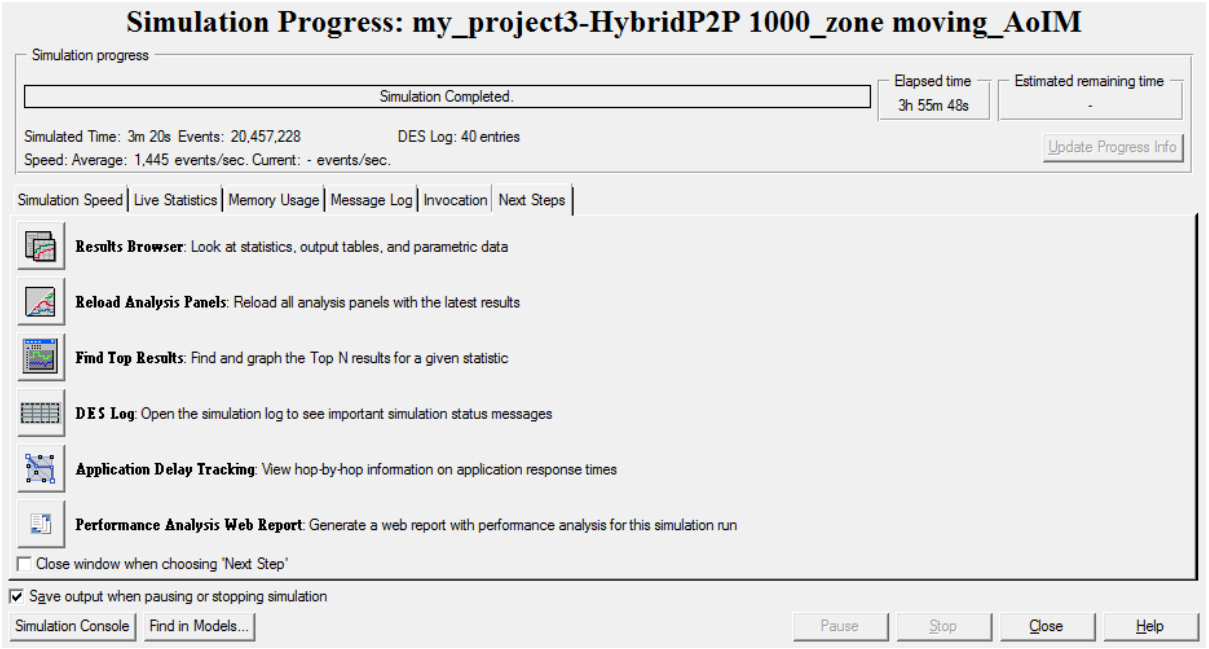


Figure 6. 28: Simulation Time (Elapsed time) for MMOGs Hybrid P2P System with 1000 peers with AoIM

6.11.2 Overall End-to-End Delay

End-to-End delay is considered one of the most important parameters in MMOGs. It is defined as the overall end-to-end delay for all packets received by the station. End- to-end delay

for the application is used during the simulation to measure the time spent from source to destination. End-to-end delay refers to the time it takes to send a packet from source to destination over a network. Figures 6.29, 6.30 and 6.31 illustrate the overall end-to-end delay results for the dynamic AoIM for MMOGs based on hybrid P2P architecture and compare the results with MMOGs based on client-server architecture. The figures below illustrate a small variation of delay when using dynamic AoIM for MMOGs hybrid P2P system with 125 and 500 players compared with the system without using AoIM. However, the variation of delay is increased when the number of players reaches 1,000 players. Nevertheless, the overall delay for dynamic AoIM of MMOGs based on hybrid P2P system is small compared to the MMOGs based on client-server architecture.

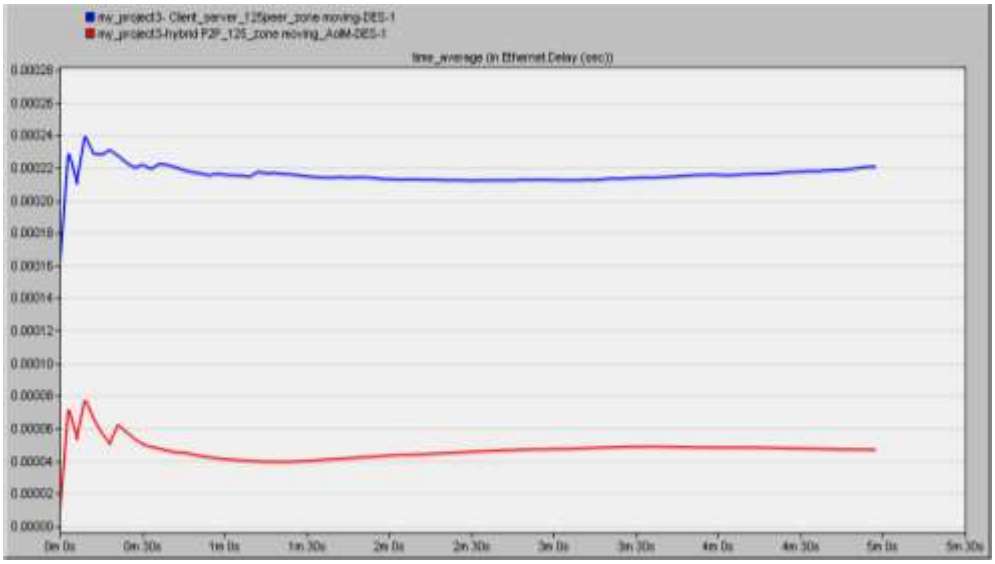


Figure 6. 29: Overall Delay for Dynamic AoIM of MMOGs with125 Peers

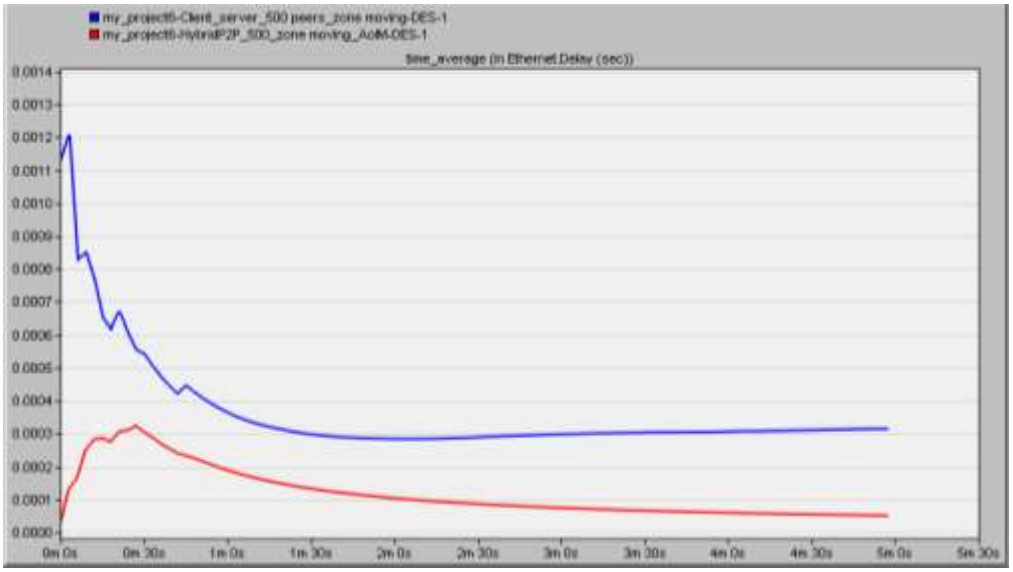


Figure 6. 30: Overall Delay for Dynamic AoIM of MMOGs with 500 Peers

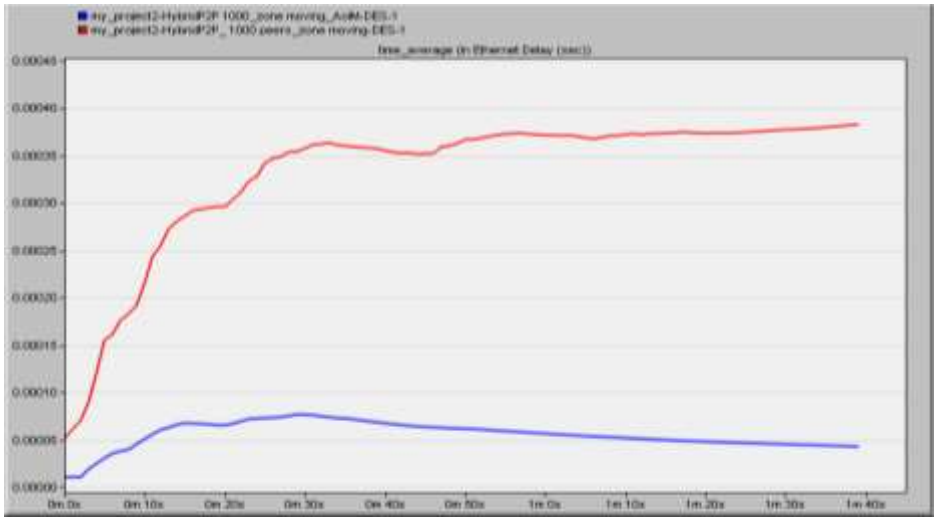


Figure 6. 31: Overall Delay for Dynamic AoIM of MMOGs with 1000 Peers

6.11.3 Traffic Received

Traffic received is considered one of the significant parameters that affect the responsiveness of MMOGs. Traffic received is the average number of bytes per second received by all the nodes in the network. In other words, network traffic is the amount of data moving across a network at a certain point in time. Network traffic is one of the key components for measuring network traffic, network traffic control and simulation. The appropriate regulation for network traffic helps to guarantee the quality of service for the network. Figures 6.32, 6.33 and 6.34 illustrate the traffic received results for the AoIM of MMOGs based on hybrid P2P and compare the results with MMOGs based on client-server architecture. The

figures below show a big variation of traffic received when using dynamic AoIM for MMOGs based on hybrid P2P system compared to the MMOGs based on client-server architecture without using AoIM. Consequently, the use of MMOGs based on hybrid P2Psystem with dynamic AoIM provides very small traffic received compared to the system of MMOGs based on client-server.

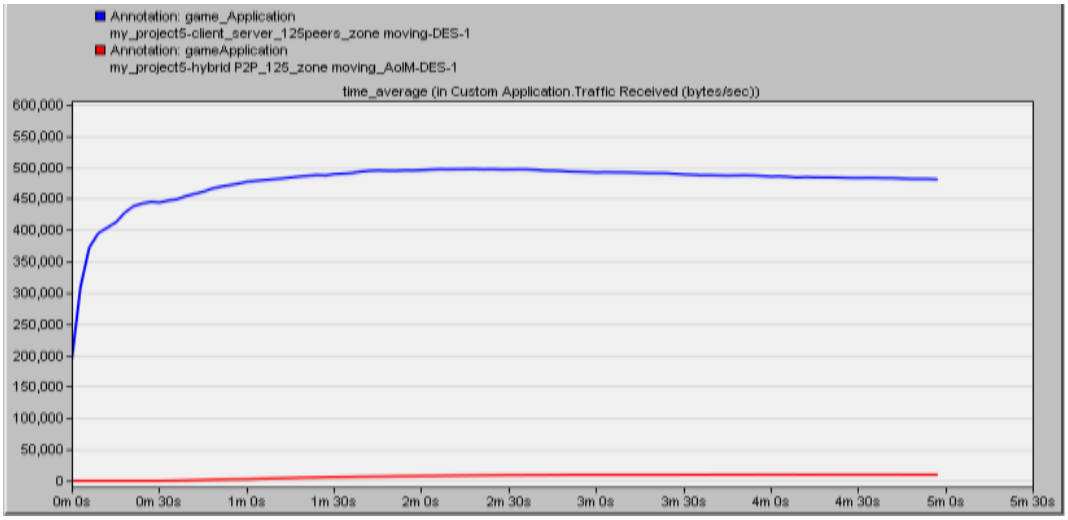


Figure 6. 32: Traffic Received for Dynamic AoIM of MMOGs with 125 Peers

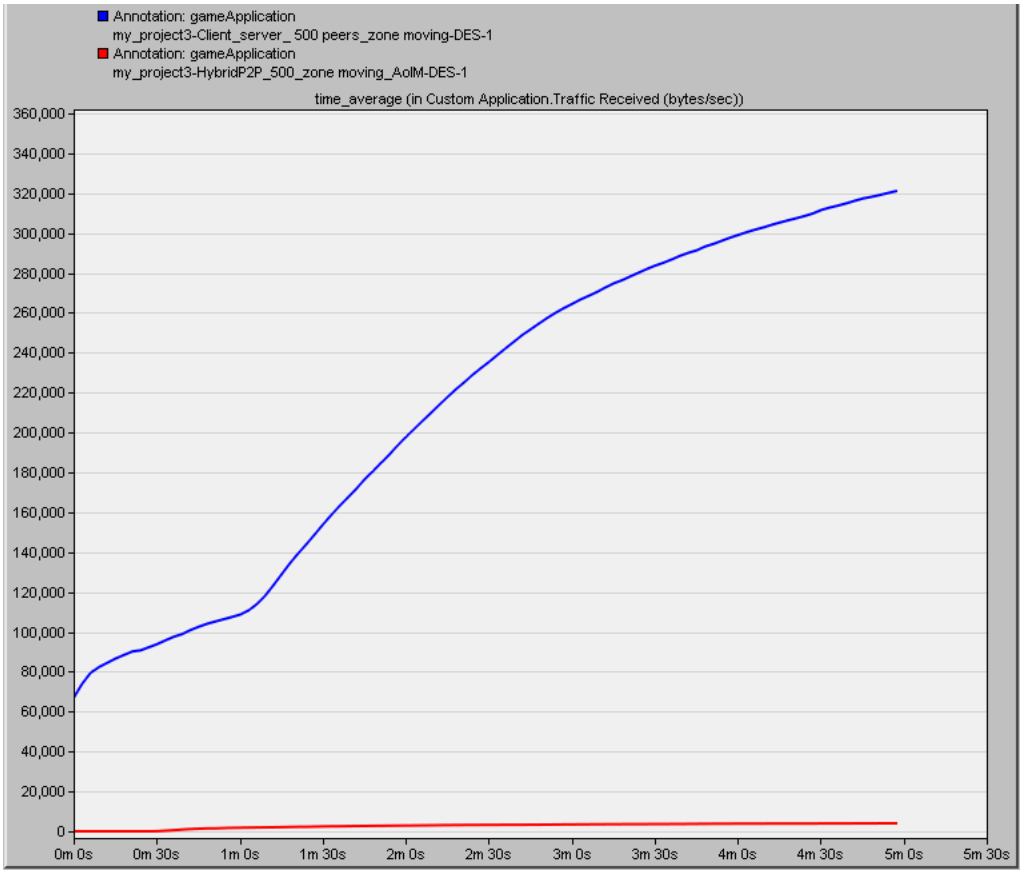


Figure 6. 33: Traffic Received for Dynamic AoIM of MMOGs with 500 Peers

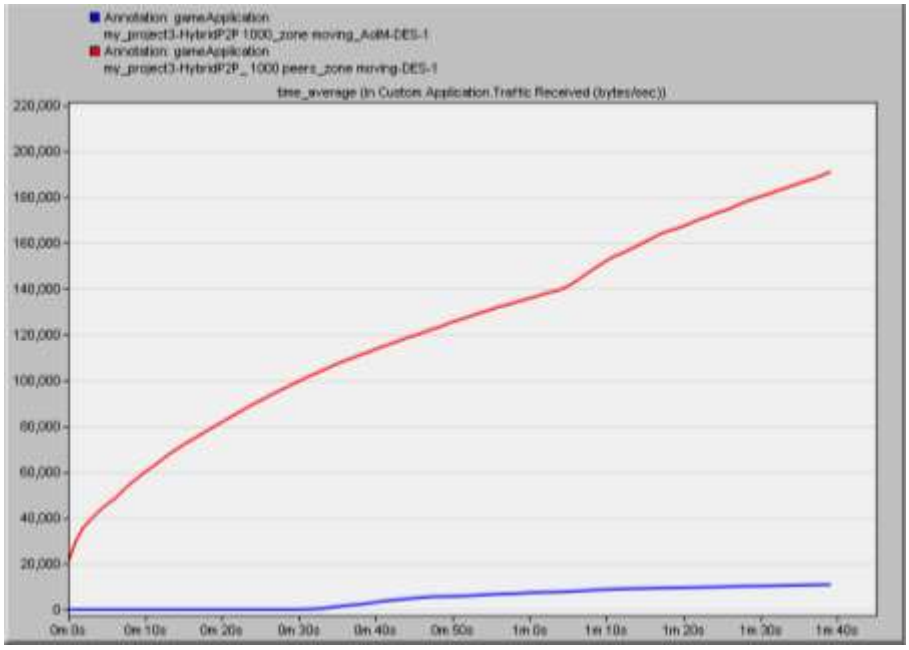


Figure 6. 34: Traffic Received for Dynamic AoIM of MMOGs with 1000 Peers

6.12 Discussion

Static and dynamic area of interest management for MMOGs have been designed and simulated using OPNET Modeler. Our dynamic AoIM is proposed to reduce the area of interest for each player in the game world space. The simulations results show that the use of dynamic AoIM for MMOGs based on hybrid P2P architecture provide a low level of delay and traffic received compared to the AoIM for MMOGs based on client-server architecture. The dynamic AoIM for MMOGs is more flexible to adapt to the rapidly increasing the player numbers. It is effective to manage the group of receivers for each player. It is considered one of the most efficient mechanisms to apply in the real world.

6.13 Conclusions

In this chapter, we have introduced the subjects and issues related to our research and explained the main contributions of the AoIM. We have also explained the area of interest management types, as well as introducing the design of both static and dynamic AoIM. We used OPNET Modeler 18.0 to model and simulate the new area of interest management system. We have simulated both kinds of AoIM and discussed the results. We have used OPNET simulation to enable the networks construction, study communication infrastructure, design individual devices, and simulate the protocols and applications. The simulation results illustrate that the MMOGs hybrid Peer-to-Peer system with AoIM scheme produces low delay and low traffic received in the network topology when compared with both MMOGs client-server and

MMOGs hybrid Peer-to-Peer system without AoIM for all scenarios used in the simulation. The static AoIM is an interesting academic exercise to highlight the benefits of the approach, however, the dynamic AoIM for MMOGs should provide a more effective technique and is more fitting with MMOGs, in which peers can be joining, leaving, failing, and migrating to other regions in the game world space.

7

STATIC AND DYNAMIC LOAD BALANCING FOR MASSIVELY MULTIPLAYER ONLINE GAMES USING OPNET

7.1 Introduction

Massively Multiplayer Online Games (MMOGs) have the possibility to support hundreds of thousands of synchronised players scattered across the world and participating with each other within a single shared game. The increase in the number of players in MMOGs has led to some issues with the demands for servers which generates a significant increase in costs for the game industry and impacts on the quality of service offered to players. With the rapidly increasing player numbers, servers still need to work efficiently under heavy load. Generally, the game server of MMOG is responsible for handling massive numbers of game players, as well as providing the indispensable consistency in order to provide the same feeling for all the game players in the same game world [89]. In real online games, when a specific game server becomes overloaded to the other games servers in the game world, it is caused by many players preferring to play in a specific region of a game world; the game server can become unsteady. This problem can be caused by an increase in network delay, traffic send and received and bandwidth. There are several research works to balance the load of the game server [90] [91] [92] but most of the researches in this area are based on the client-server architecture. In this chapter, we have proposed and designed a new static and dynamic load balancing technique for MMOGs based on hybrid P2P architecture [93].

7.2 Load Balancing in P2P MMOGs

Load balancing can be defined as the process of effectively distributing processing requirements of applications across a number of various servers [72] and is the main concern for all distributed systems. In other words, load balancing technique is linked to the mechanism to distribute the load of processing that occurs when the peers join and leave the system. Actually, the scalability problem in MMOGs based on Client/Server architecture is intrinsically related to load balancing [94]. However, when applying load balancing for MMOGs' infrastructure, it must be implemented in an effective way to avert essential resources over-supplying on the server side. The basic method to load balancing is to migrate nodes from heavily loaded to the other lightly loaded and then to redistribute the load across the nodes. However, this load balancing approach is far from straightforward in a peer-to-peer system. In P2P architecture, there are two main issues. Firstly, how to determine that the node is overloaded or under-loaded. Secondly, how to find an appropriate partner node where to redistribute the load [95]. In [96], Naaz et al., presented three essential parameters which generally define the strategy a particular load balancing algorithm will employ. These parameters are the maker of the load balancing decision, the information used to make the load balancing decision, and the time of the load balancing decision being made. Load balancing can be categorised into two main types: static and dynamic.

7.3 Dynamic Load Balancing Model

7.3.1 Definition of the Load

In order to better understand the load balancing technique, there are three types of node's state load: light load, moderate load, and heavy load. The node (server) called light load (idle) when the node's resources (CPU, memory, bandwidth, and disc storage) are lightly utilized. Correspondingly, the node is heavy load when the node's resources are highly utilized. However, the moderate load means there is normal utilization of the node's resources.

7.3.2 MMOG Server Load Characteristics

There are several critical factors that influence the server load in MMOGs. These should be taken into account when we want to design the load balancer for MMOGs [97].

The first and most important factor is the process load in the game server. It is known as the work load, it is generated due to the process and management of entities. The second factor is called the communication load, it is generated from the interaction of entities in

boundary zones of clustered servers. The final factor is called management load, it is generated from domain management in the virtual game world. Management load does not dependent on the numbers of users. However, it is a default load generated by the allocated game space. It is a quite small load compared with the work load and communication load. The boundary domains for each region should be determined before the start of the game so that the average communication load can be reduced in boundary domains of the game world. MMOGs servers' work load and management load are absolutely generated and proportionally depending on the numbers of players as well as the server area scale. However, the communication load is varied according to the boundary area of the game world.

7.3.3 Load Balancing Problem in MMOGs

One of the significant aspects in MMOGs is the game performance if the response time of the game request is reduced. The load balancing problem becomes more important when the number of MMOGs players is rapidly increased. The problem of balance and distribution of the load of MMOGs based on hybrid P2P architecture are very important and challenging area of research. The major problem of the load balancing in MMOGs based on hybrid P2P system is to determine the optimal way to achieve a balance and distribute the load among the regions on the game world space. Load balancing is considered one of the most substantial problems for attaining a high level of performance in MMOGs. With the increase in player numbers and the possibility of migration from one region to another inside the game world space, it is possible for some regions to become overloaded while the others are lightly loaded. This can lead to a sharp decline in the level of performance [98].

7.4 Load Balancing Techniques

Load balancing techniques are primarily used for balancing the workload in distributed systems. The load balancing can be classified into two main types: "static load balancing" and "dynamic load balancing". These kinds of load balancing are explained in the next sections. The main objective of load balancing techniques is to improve the level of performance by redistributing the load between available server nodes. Dynamic load balancing techniques react to the current state of the system, while static load balancing techniques rely on just the average system behaviour in order to achieve the load balancing of the system, transfer decisions are separated from the actual current state of the system. This situation makes the dynamic technique more complicated than static one. However, dynamic load balancing policies have the possibility to achieved the best performance compared to static load balancing

[99]. The performance of different load balancing techniques is measured by several parameters such as fault tolerant, centralised or decentralised, scalability, reliability, stability, migration, resource utilization, and responsiveness [100].

7.4.1 Static Load Balancing

Static load balancing algorithm works statically without the need of the present state of nodes. Static load balancing algorithm basically depends on the information about the average of the system work load without the need for the actual current system status. The performance of the processors is specified during the compilation time. Then according to their performance, the master processor is responsible for assigning the work load. However, the slave processors are responsible for calculating their assigned work and submitting the result to the master processor. A task is always implemented on the processor to which it is allocated, that is static load balancing algorithms are non-pre-emptive. The static load balancing algorithms work to distribute load according to a fixed group of rules, and these rules are linked to the kind of load, such as CPU power requirement and memory requirement [98]. All the prior information of the system is previously known, such as CPU power, memory availability, performance as well as data about node's requirements for instance bandwidth. Static load balancing algorithm works with less complex situations, the reason is it does not need the information relating to the present state of the system [101]. This type of load balancing has serious disadvantages when the sudden failure happens in the system resources, tasks cannot be moved during its implementation for the load balancing. Round robin is considered one of the models of static load balancing algorithm which divides the traffic evenly between servers. However, there are several problems appearing in round robin algorithm, in order to cope with these problems, a new load balancing algorithm is proposed called Weighted Round Robin [101]. The major idea for this algorithm is that each server has been allocated a weight, therefore the servers that have the highest weight will receive more connections. The main objectives of static load balancing algorithm are to decrease the execution time of the processes and reduce the communication delay [100]. However, the main disadvantage of this technique is the algorithm does not check the load of other nodes in the network. Thus, they cannot guarantee whether they balance up or not. This leads to decrease the performance of the system.

7.4.2 Dynamic Load Balancing

In this approach, the processes are assigned to different processors based on the new information collected [102]. Dynamic load balancing allocates processes dynamically

when one of the processors becomes under loaded. Dynamic load balancing can provide a considerable improvement in performance when compared with static load balancing technique. The static load balancing algorithms are more stable compared to dynamic algorithms. When one of the processors becomes under loaded, dynamic algorithms can be allocated processes dynamically [103]. However, this approach provides an efficient load balancing but with additional cost for collecting and maintaining load information, thus it is significant to maintain these overheads within sensible limits [102]. This method is suitable for MMOGs because of the changing resources of the game world, also changes in avatar behaviour.

Dynamic load balancing algorithms work on present state of node and distribute the load between the nodes at run time. The decision of balancing is taken according to the present status of the system. Dynamic load balancing is implemented, when the load of the system and processes number is probably to change at run time. In this situation, there is a need for permanently monitoring the load of the system. This case will increase the overhead and makes the system more complicated compared to the static policies [104]. Substantially dynamic load balancer is used to track real time load distribution across the system, and all the decisions will be taken dependent on the system wide load. It also makes sure that the load is evenly distributed across all nodes.

One of the key purposes of dynamic load balancing is to decrease the load produced from the game by equally controlling and real time monitoring the loads between servers (super-peers) in the game world. In other words, the load balancing is used to distribute load among a number of nodes in order to improve the benefit of the computation capability of each node, as well as to achieve optimum resource utilization which maximises throughput and minimises the task response time. Load balancing is very fundamental in MMOGs in order to improve the quality of service by dealing with continuous changes of the load over time which leads to improvement of the performance of the system. The load balancing leads to a reduction in the overall waiting time of the resources and averts too much loading on the node's resources. The main advantages of using load balancing is to increase the availability of resources, improve the performance of the system by increasing reliability, increase the throughput, maintain and increase the level of stability, optimize resource utilization and provide fault tolerance ability [99].

There are two main methods for dynamic load balancing: distributed and non-distributed [104]. In the distributed dynamic load balancing, the load balancing is done for all nodes in the system and the load balancing task is shared between them in the network system. Every node communicates with all the other nodes in the system. This leads to a high level of internal process communication. The benefit of using this method is to give a good fault tolerance for the system. When one or more nodes in the system fail, it will not affect the whole load balancing process. This can improve the performance of the system.

However, in non-distributed dynamic load balancing, either one node or a set of nodes do the load balancing task. It can be described in two forms: centralised and semi-distributed. In centralised form, just one master node implements the load balancing in the whole system such as central node (server). The other remaining nodes communicate just with the master node. However, in semi-distributed form, all system nodes are divided into several clusters. Each cluster is working in centralised form. Election technique is used to select a central node for each cluster. The load balancing of the system is implemented by using the centralised nodes. Obviously, the dynamic load balancing is more complex compared to the static technique, but the dynamic technique has a priority rather than static technique due to the better level of performance that is provided by the dynamic load balancing. Also the interaction between nodes to achieve load balancing can be divided into two main types: cooperative interaction and non-cooperative interaction [104]. The cooperative nodes interaction are working side-by-side to achieve the load balancing. The main goal is to improve the overall level of system performance. However, in a non-cooperative interaction, each node in the system works independently to achieve its own goals. This methods is used to improve the response time of a local task

7.4.3 Load Balancing Algorithms Types

Load balancing algorithms can be classified into three kinds according to the processes initiated: Sender Initiated, Receiver Initiated, and Symmetric Initiated. These types explained in the following section [98].

- **Sender Initiated**: In this kind, the load balancing algorithm is initialised by using the sender. The overloaded node search for the lightly loaded node to accept the load .The sender sends a request message and waits for a receiver to accept the load.

- **Receiver Initiated**: In this kind, the load balancing algorithm is initiated by using the receiver. The lightly loaded node searches for an overloaded node to get the load from it. The receiver sends request messages and waits for a sender that can get the load.
- **Symmetric Initiated**: This type is an integration of both sender initiated and receiver initiated according to the current system state.

7.5 Qualitative Measurement

The performance and qualitative measurement of different load balancing techniques is measured by using several parameters. These parameters are described in the following section [105].

7.5.1 Response Time

It refers to the amount of time spent by the load-balancing technique to reply the distribution of work load between the nodes in the system. The Static load balancing algorithms take less time than dynamic load balancing algorithms. The reason is due to the dynamic load balancing taking a long time to make the redistribution decision of the processes between the processors.

7.5.2 Fault Tolerant

It refers to the evaluation of load balancing algorithm performance in the case of failure in order to verify if the performance of the system declines when one or more nodes fails or not. Dynamic load balancing algorithms have the capability to cope with failure compared with the static load balancing algorithms because they support process migration.

7.5.3 Process Migration

It indicates whether the transfer of the load occurs inside the nodes of the load balancing system or not. This parameter is able to decide whether it should redistribute the load during the process implementation or not.

7.5.4 Overload Rejection

It is related to the determining of the overload amount involved during executing the load-balancing technique. It is included in the overload because of the movement of workloads, inter-process communication and inter-processor communication.

7.5.5 Reliability

It relates to the ability of load balancing algorithm to carry on in stellar way when one or more nodes are failed. Dynamic load balancing algorithm provides a better level of reliability than the static load balancing. The reason is that there is no task/process that will be moved to another host in case a node fails at runtime in static algorithm.

7.5.6 Adaptability

It is used to verify whether the load balancing algorithm has the ability to be adaptive with varying cases. Dynamic load balancing algorithms are adaptive. However, static load balancing algorithms are not adaptive.

7.5.7 Stability

It refers to the delays in the movement of information among processors as well as the benefits of using the load balancing algorithm which leads to achieve a fast performance within a specified amount of time. Static load balancing algorithm provides a better level of stability than the dynamic load balancing. The reason is that there is no information exchanged between processors during the run time.

7.5.8 Resource Utilization

It refers to the resources that are used in the server. Dynamic load balancing has relatively preferable resource utilization because it takes into account the fact that the load should be evenly distributed to processors, so that eventually no processors will sit idle. However, static load balancing has minimal resource utilization because it only tries to allocate tasks to processors in order to achieve less response time, ignoring the case of some processors which complete their work early and sit idle because of lack of work.

7.6 The Requirement for Designing a New Load Balancer

1- The load balancer must be capable of measuring the current load work for each server according to CPU utilization, CPU queue length, idle time percentage, the amount of incomplete work, and the requirements of resource for each job.
2- The load balancer must be capable of determining the functionality of each server in the network system.
3- The load balancer must be capable of guiding a user's requests to the right server for the period of the user's session.

4- The load balancer must guarantee to respond to all the user's requests in the same network system.
5- The load balancer must be scalable to the increased number of players in the game world space. It means the system can serve a single player, as well as hundreds or occasionally thousands of concurrent players.

7.7 OPNET Process Domain

Process models in OPNET Modeler are used to determine the behaviour of both processor and queue modules which exist in the node domain. Process models are used to execute a wide selection of hardware and software subsystems, for example communication protocols, algorithms, operating systems, custom statistic collectors, and shared resources such as disks and memory. The process model operation is explained in the following section [106].

7.7.1 OPNET Process Model Operation

Node domain is used to represent devices that can be linked together to form networks. Network devices can be ranged from simple traffic sources to complex switches or sometimes computers running multiple applications. Nodes are formed as examples of node models, a node model is the schema for all individual nodes of a certain type. OPNET node models are defined as a group of modules representing distinct functional areas of the node. Particular modules are restricted in the kinds of behaviour they can represent; for instance, the several transmitters and receivers responsible for representing the interfaces to connect defined in the network domain. In OPNET Modeler, there are three kinds of node modules processors, queues, and esys modules that support completely general behavioural modelling. These modules provide basically the same abilities in relation to their general conduct and most of their physical resources. Processors, queues, and esys modules can be handled identically because they depend on the same process modelling technology to represent their conduct. Subsequently, in this chapter the term QP is used to represent these three modules [106].

7.7.2 OPNET Process Environment

Individual or sets of processes are defined to execute a special task when placed inside a QP [106]. A process is an example of a process model that is defined with the Process Editor. All individual process examples are defined to have the same characteristics but may operate in various environments in OPNET Modeler. Process models are responsible for executing one

of two functions, relying on the QP. A process model usually represents a behavioural model of a process, as demonstrated below. Esys modules can also use a process model to deal with interaction with the external system. Effectively, the process model becomes part of an interface between Modeler and the external system during a co-simulation. This next section explains in general terms the context of process model components.

7.7.3 OPNET Process Model Components

OPNET Process models are considered to be the most important specifications for the dynamic behaviour of processes [106]. A process model must describe all the actions that a process will execute under all potential circumstances. In several situations, all the processes that are implemented under discrete event simulations represent a real-world equivalent algorithm or protocol. The official specification of both protocol and algorithm is a higher-level specification than the requirements must be met by executing a process that is used in a standardised manner. A standard specification describes all the requirements that apply for processes of the real world, as well as to the models belonging to those processes. The example of standard specification is the IEEE 802 for the local area network protocol, and the Reference Documents (RFC documents) for the TCP/IP protocol suite. Actually, any group of requirements that apply to the process actions and interactions can be named a standard for the executions of that process.

7.7.4 OPNET State Transition Diagrams

The programming language that is used to develop the algorithm in OPNET Modeler is called Proto-C model. Proto-C model consists of two fundamental component types: states and transitions, thus the name state transition diagram (STD) [106]. States are typically used to represent the highest level of modes that a process can enter. While, transitions are used to determine the changes in state that are potential happened for the process. STDs are graphical pictorial in the Process Editor. However, there are other significant sides in Proto-C process models that are not graphically represented. These contain actions related to each state, variable and attribute announcements, and common definitions of functions and expressions. The next section describes the abilities and applications of these features.

7.7.5 Forced and Unforced States

Generally, the word state refers to the information that a process might have accumulated over the time. However, the process of state perhaps does not contain all the

information obtained by the process, but just the information that it has selected to maintain. Also state information may be on an ongoing basis updated when new events happen and data becomes available. In STD, the term state refers to an object that corresponds to one of the basic modes or cases that a process might find itself in. States are mutually equivalent and integrated, this mean that a process is always in precisely one state: more than one state might not be occupied at a time. The transitions that leave from a state refer to the states that can be occupied next, as well as the requirement for each change.

Actions specifications may be related to each Proto-C state. In Proto-C environment, all the actions are called executives. The state executives are divided into two parts: enter executives and exit executives. As the names suggest, a state's enter executives are implemented when a process enters the state, while exit executives are implemented when the process leaves to follow one of the outside transformations. In Proto-C module, there are two kinds of states: forced and unforced states that differ in implementation-timing. Forced states are graphically represented as green circles (State 2), while unforced states are represented as red circles (State 1) as shown in figure 7.1.

Unforced states [106] give permission to pause between the enter executives and exit executives, and therefore can be modelled a true states of a system. In unforced state, when a process has finished the enter executives, it blocks and returns the control to the former context that invoked it. At this point, the process stays outstanding until a new invocation causing progress to the exit executives of its current state. Figure 7.2 illustrates the flow of implementation using the unforced states of an STD as interrupts cause the process to move forward.

Unforced State Forced State

Figure 7. 1: OPNET Forced and Unforced States

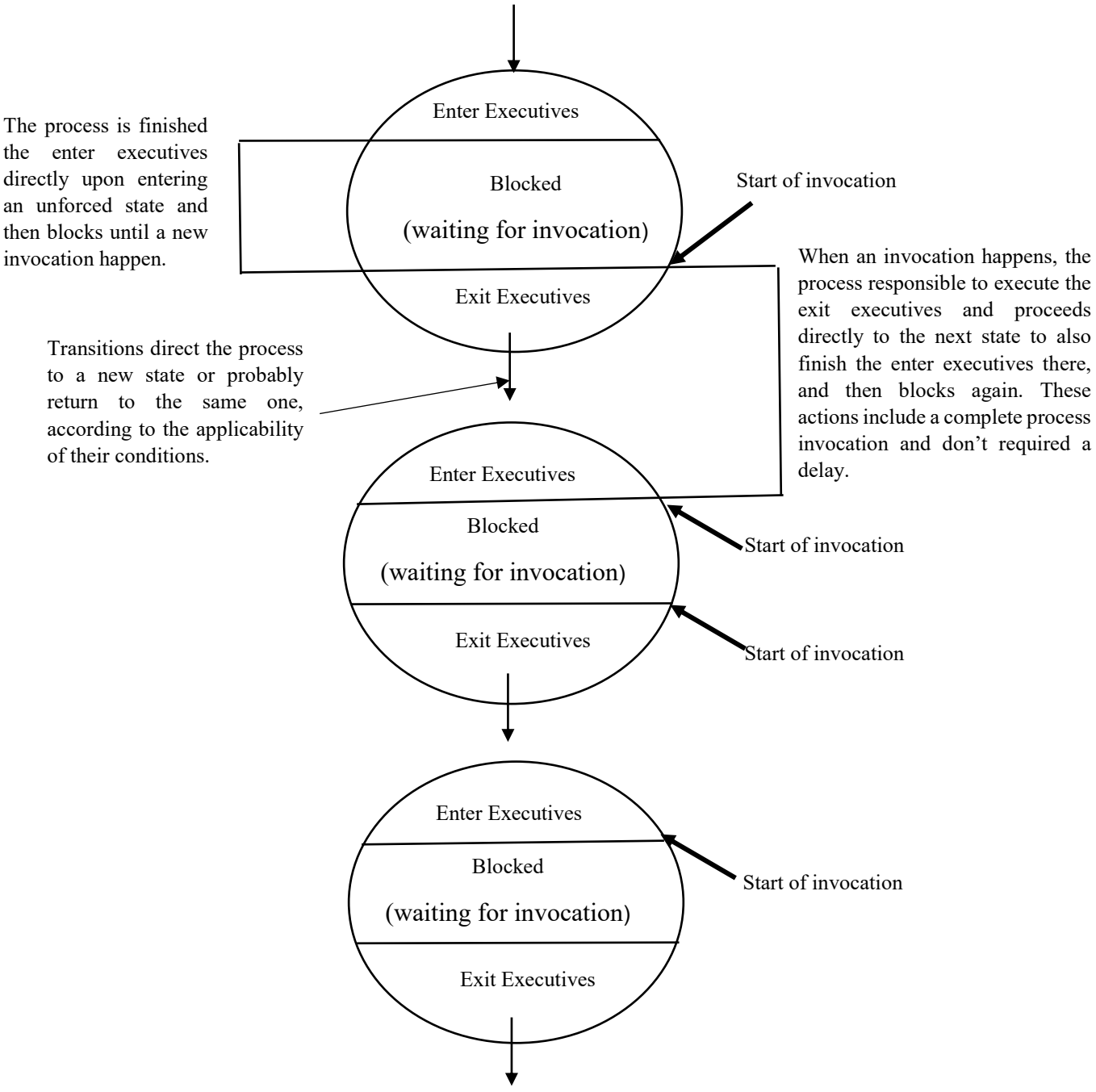


Figure 7. 2: OPNET Implementation Flow using Unforced States

In forced states [106], they do not allow the process to wait. Therefore they cannot be used to represent the system that continues for any duration. The exit executives of a forced state are implemented by a process directly upon finishing of the enter executives. Commonly, the exit executives of a forced state are left blank, because they have exactly the same statements that developed at the end of the enter executives. Typically, forced states are not used as much as unforced states because it does not represent the actual system states.

Nevertheless, they are beneficial in particular situations to separate actions graphically or control over the flow of decisions which are common to various unforced states. As for the graphical separation of decisions or actions, it can occasionally provide better stereotyping specifications in addition to further visually informative state transition diagram. The used of forced states only caused a small modification in the implementation flow illustrated in figure 7.2.

7.8 Design Dynamic Load Balancing Technique for MMOGs

Dynamic load balancing techniques rely on recent system load information in addition assigning the job for each server node at run time. The load balancing decisions are dependent on the current system state. Thus allowing the workload to dynamically shift from an overloaded node to the light-load node in order to increase the response time of server nodes. The main advantage of using the dynamic load balancing is the capability of responding to difference in the system. We have designed a new dynamic load balancer for MMOGs based on hybrid P2P system. Our load balancing system measured the load of each super-peer in the game world space in order to find out the availability for each region. The measurement of the load is based on the node resources such as power of the CPU, memory available, bandwidth, disk space as well as stability and reactivity. When the first player joins the game for the first time, it must be registered in the game server. The server will assign this player as a super-peer for the region. After that, each player wanting to join the game must be registered in the game server. The game server sends the ID of the player to the dynamic load balancer to find the suitable region to connect to it. The load balancer is responsible for sending a message to the super-peer of the region to inform him that a new player will join the region. When the player joins the second time or subsequent times, it should join the game through the game server in order to find the appropriate region to allocate to it. However, in the case of a player leaving the game, it should inform the super-peer or clone-super-peer of the region about that. The super-peer is responsible for informing both the dynamic load balancer and the game server. The dynamic load balancer updates the load of each region when the new player joins the game or when the player leaves the game. When the super-peer of the region leaves the game, it must inform both the game server and the clone-super-peer of the region about that. The clone-server will become the super-peer of the region and assign a new clone-super-peer for that region. All the information will be updated in the game server and the dynamic load balancer. If all the regions in the game world space are overloaded, the load balancer will inform the game server to create a new region in order to handle the work load of the players. While, if there are regions

that do not contain any players, the dynamic load balancer will work with the server to destroy the region and allocate the super-peer and clone-super-peer of that region to another convenient region in the game world space. The dynamic load balancing algorithm is explained in the next section.

7.8.1 The Algorithm of Dynamic Load Balancing for MMOGs based on Hybrid P2P system

The dynamic load balancing algorithm is used to distribute the load of players over different regions in the game world in order to achieve the great performance and high level of scalability with low delay and traffic received as the game world environment is heterogeneous.

The load balancing system consists of the following elements periodically sent by each region's super-peer.

- **Game server**: The game server is responsible for receiving several types of load information. This information will be sent to the load balancer assistant.
- **Game Load Balancer**: it is the main part of the load balancing algorithm. According to the load information for each region, the load balancer will send the login's player to the suitable region in the game world space.

Input
N← Max number of nodes
S ← Centralised game server
R ← Number of regions

Output
Number of regions with light weight of load with suitable number of node in each regions and each region is controlled by using both super-peer and clone-super-peer.

1.	Each region's super-peers send the load information to the game server
2.	Each node starts the game by registering in the game server.
3.	All the node information is stored in the game server database in order to use this information when the player returns to play again.
4.	Calculate the load of each region according to these criteria: o Number of players available in the region. o High performance super-peer ▪ *Super-peer disk capacity.* ▪ *Super-peer memory* ▪ *Super-peer CPU power* ▪ *Super-peer bandwidth* ▪ *Stability* ▪ *Availability*
5.	According to the load information for each region, the load balancer will decide to locate the node to the suitable region.
6.	The game load balancer sends a message for both game server and region super-peer to inform them about this new node.
7.	The game load balancer sends a message for both game server and region super-peer to inform them about this new node.
8.	The super-peer and clone-super-peer of the region control and manage the movements of the player from one region to another.
9.	When the node is leaving the game, it informs the region super-peer.
10	The region super-peer sends a message for the game server about this leaving.
11	The game load balancer updates the load information for this region.
12	If the super-peer is leaving the game, it must inform both clone-super-peer of region and the game server about that.
13	The clone-super-peer will be the super-peer of the region and choose the suitable node to be a clone-super-peer.

7.8.2 Explanation of Dynamic Load Balancing Algorithm

Each player must be registered in the game server at the beginning of the game. The first player will be assigned as the super-peer of the first region in the game world space.

However, the second player will be allocated as the clone-super-peer of that region. All the information of the super-peer and clone-super-peer will be sent to the load balancer in order to inform it. After that, each player will be assigned as a normal player in the region by sending all the players from the game server to the super-peer through the load balancer. When the first region is full, the load balancer is responsible for informing the game server about that in order to create a new region and assign both super-peer and clone super-peer for that region. If the normal player wants to leave the game, he/she will inform the super-peer of the region about that leaving. The super-peer turn to inform both load balancer and the game server to update the load of this region. However, if the player decides to migrate to another region, he/she will inform the clone-super-peer and send the region ID. The clone-super-peer will check the availability and the possibility of that region to join the player through the load balancer. If the new region is available and possible to accept the player, the load balancer is responsible for allocating the player to that region and update the load information for the two regions. All the updated information is sent to the game server and the super-peers of the regions concerned. As for the leaving or migration of the super-peer of the region, he/she will send a notification the game server, load balancer and clone-super-peer of the region. The clone-super-peer will be assigned as the super-peer of this region and he/she has the possibility to allocate a new clone-super-peer for the region. All the updated information is sent to the game server and the load balancer. However, if the clone-super-peer leaves or migrates the game, he/she will inform the super-peer of the region to assign new clone-super-peer of the region. All the updated information is sent to the load balancer and the game server.

7.8.3 Simulation of Proposed System

We have used simulation in order to evaluate the proposed system. We have designed a new load balancer by using OPNET Modeler programming language Proto-C. The process model and node model will be explained in the following section. As for the leaving or migration of the super-peer of the region, he/she will send a notification to the game server, load balancer, and clone-super-peer of the region. The clone-super-peer will be assigned as the super-peer of this region and he/she has the possibility to allocate a new clone-super-peer for the region. All the updated information will be sent to the game server and the load balancer.

7.8.4 OPNET Process Model and Node Model for our Load Balancer

We have used the process editor to design the process model for the dynamic load balancer for MMOGs. Our model consists of three states, two forced states and one unforced

state as shown in the figure 7.3. The first state is called INIT state. It is used for initialisation of the load balancer such as configuration, policies, and the availability of the system. The second state is called idle. It is used to represent a waiting station of the packets, as well as a loop to send the packet to the process state. The third state is called process_pkt. It is used to represent the core of the load balancer process. The node model of the dynamic load balancer consists of two processors called load balancer pro and traffic source, in addition to one receiver and one transmitter. The load balancer pro processor is used to execute the process model of the load balancer, while the traffic source is used to generate the traffic to send it to the load balancer pro. All the processors and the transmitter and the receiver are connected by using the packet stream link. We have used the logical Tx/Rx association between the receiver and transmitter in order to ensure the correct transmitter and receiver pair are used when connecting links to node at the network levels as shown in figure 7.4.

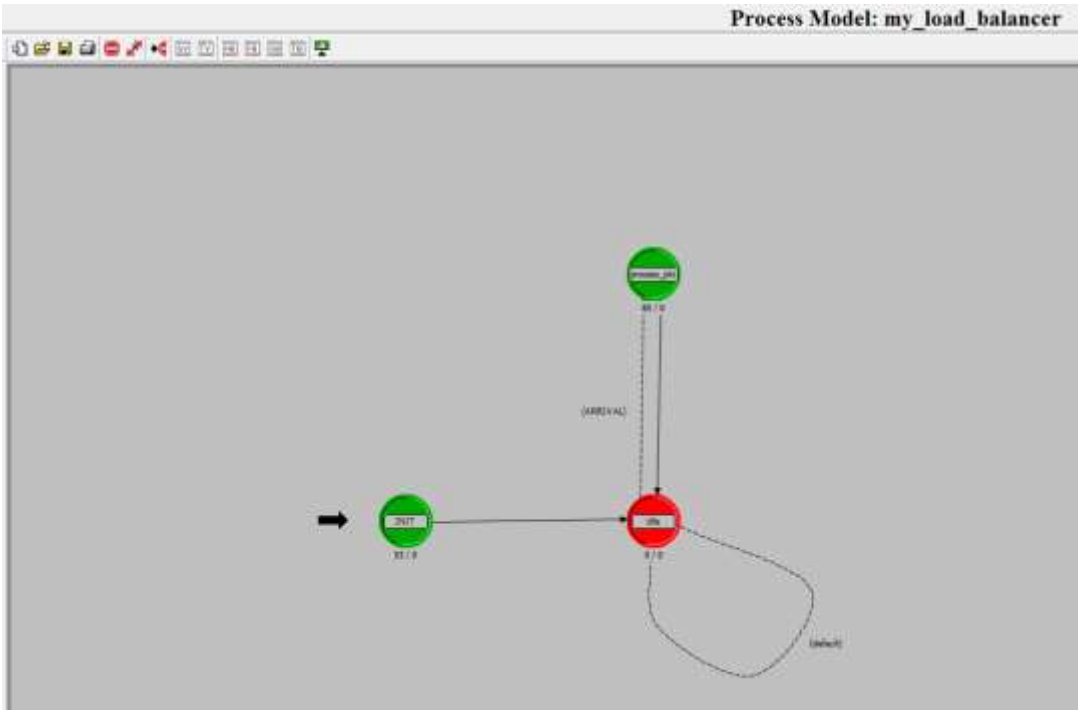


Figure 7. 3: Dynamic Load Balancer Process Model

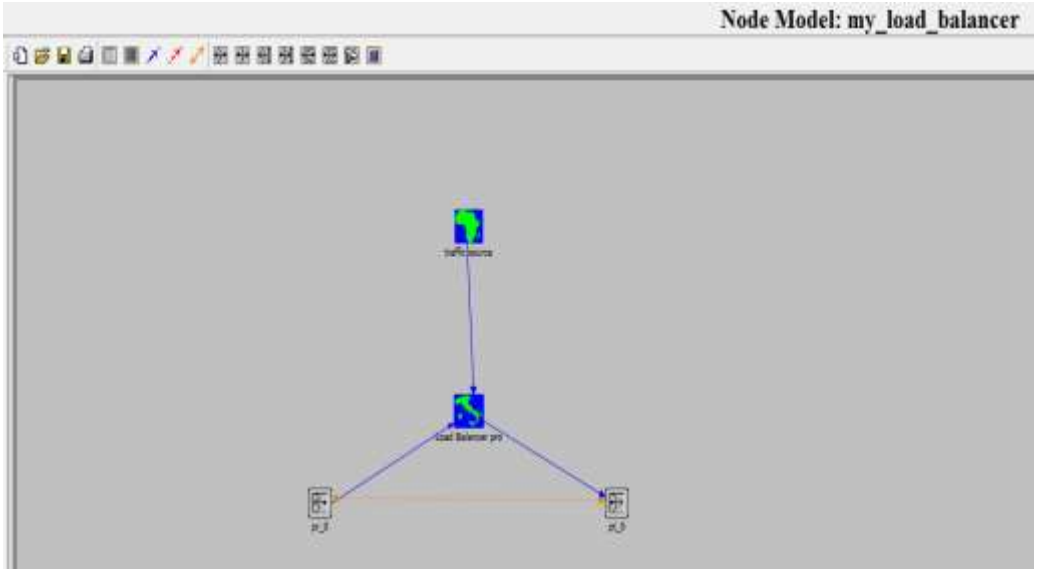


Figure 7. 4: Dynamic Load Balancer Node Model.

7.8.5 Failure and Recovery Mechanisms

Each processor module in OPNET Modeler has several attributes that allow the Simulation Kernel to determine the expectations about receiving failure and/or recovery interrupts. The failure attribute interrupts refer to the module that will expect to receive failure interrupts, and the same thing for the recovery attribute interrupts refer to the module that will expect to receive recovery interrupts. There are three types of interrupt attributes [107]: network wide, local only and disabled. The network wide attribute indicates to the module that it will receive an interrupt if any node or even link fails or recovers, regardless of the location in the network. While the local only attribute indicates to the module that it will receive an interrupt just when its parent node fails or recovers. However, the disabled attribute indicates to the module that it will not receive any failure or recovery interrupts. Disabled is referred to the default value for the attributes, because most of the modules usually do not expect these interrupts.

There are several related actions taking place when a node failure happens. The Simulation Kernel comments on the current event and delivers the failure interrupts to the right modules everywhere in the model. After the failure interrupts are delivered, the Simulation Kernel is responsible for informing the node's transmitter and receiver channels of the failure. The transmitter channels will abort their current packets in progress and also flush their transmit queues. The receiver channel assign the node failure (Transmission Data Attribute

(TDA)) in its packet(s) to the present simulation time. In the transceiver lines, a packet will not be accepted when the node fails at any time through its transmission or reception. After the interrupts are delivered and the transmitter and receiver channels have completed their actions, the value of the Condition attribute is internally assigned to disabled and the suspended event is reactivated. After the node recovers from the failure, the value of the Condition attribute is automatically assigned to enable and then a series of actions occur. The Simulation Kernel is responsible for suspending the present event and delivers the recovery interrupts to the right modules. After the recovery interrupts are delivered, the Simulation Kernel informs the node's receiver channels of that recovery. When the interrupts have been delivered and the receiver channels have finished their works, the suspended event will be reactivated.

7.8.6 Generating Failures and Recoveries

The node in OPNET Modeler cannot autonomously fail or recover, because an external action must be taken to alter the node state during the simulation. There are four techniques that are used to generate failures and recoveries of nodes: deterministic, scripted, stochastic and interactive. The deterministic technique is used to cause one node to fail and recover at specified times during simulation. While the script technique is considered more general than the deterministic technique and it is depending on the file that contains times and locations of failures and recoveries. However, the stochastic technique is used to generate failure and recoveries depending on the specific distribution of times and locations. The interactive technique is used to generate a failure and recovery directly by using the OPNET debugger. A failure or recovery happens for a node when the condition attribute of the node changes value. Thus, each technique of the four above provides a suitable mechanism to modify the attribute's value, but they are various in the manner in which the node, as well as the time for both failure and recovery, is selected.

7.8.7 Simulation of Static Load Balancing

The static load balancer system is simulated by using OPNET Modeler 18.0 [43]. In static load balancing for MMOGs based on hybrid P2P, we have used TDMA device to represent the player inside the game world space. The benefit of using TDMA is to design a dynamic AoIM for the game world as described in the previous chapter. However, the static load balancing for MMOGs based on client-server used the wireless LAN workstation to represent the player inside the game world space. All the players' nodes in both architectures have the possibility to move from one region to another in addition to move inside the region.

The position of the load balancer is between the game server and the regions as shown in the figures below. Figures 7.5, 7.6, and 7.7 show the static load balancing for MMOGs based on hybrid P2P architecture. However, figures 7.8 and 7.9 show the static load balancing for MMOGs based on client-server systems.

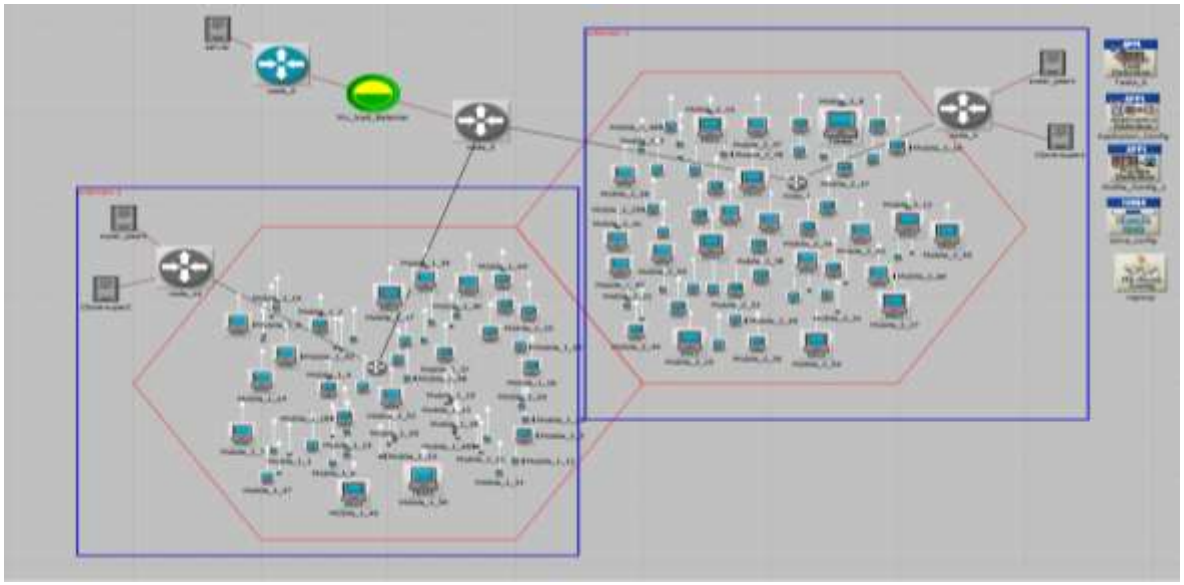


Figure 7. 5: Static Load Balancing for MMOGs based on Hybrid P2P System with 125 Peers

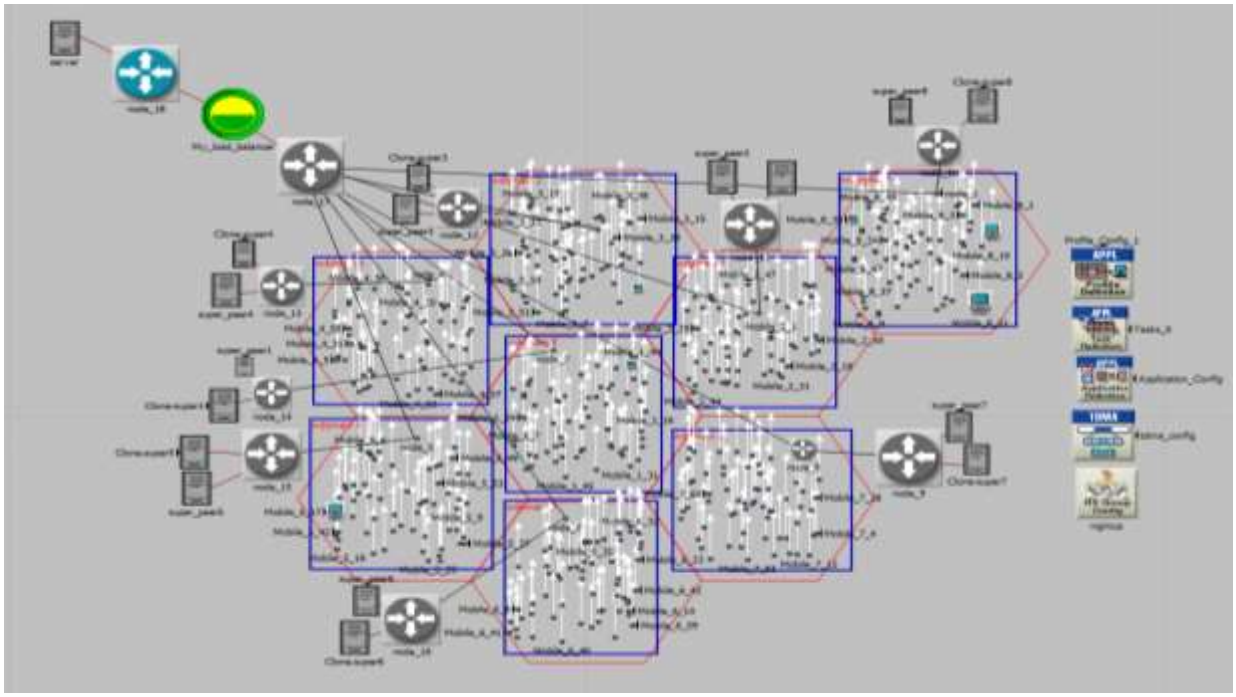


Figure 7. 6: Static Load Balancing for MMOGs based on Hybrid P2P System with 500 Peers

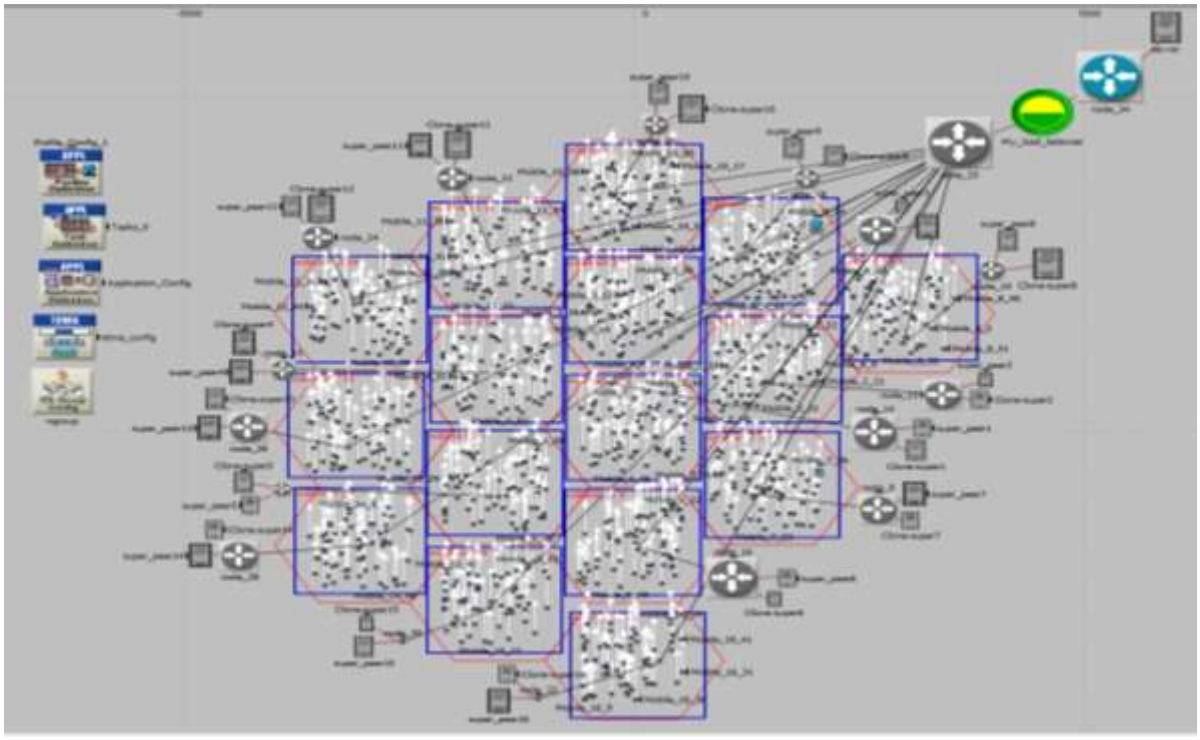


Figure 7. 7: Static Load Balancing for MMOGs based on Hybrid P2P System with 1000 Peers

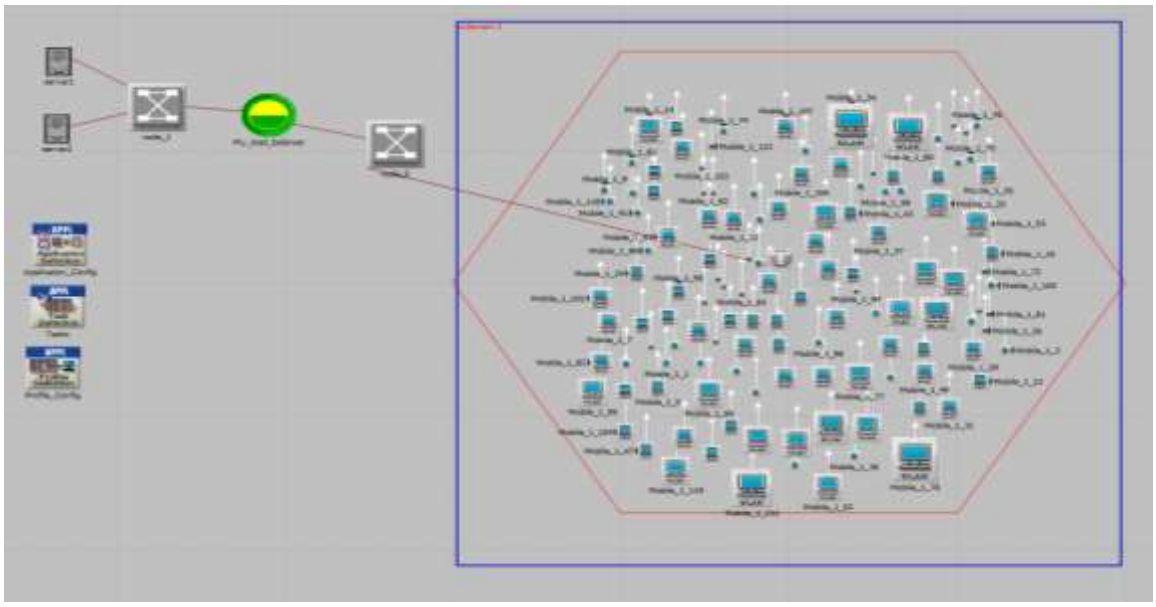


Figure 7. 8: Static Load Balancing for MMOGs based on Client-Server System with 125 Nodes

Figure 7. 9: Static Load Balancing for MMOGs based on Client-Server System with 500 Nodes

7.9 Results of Static Load Balancing

In this section, we have analysed the network performance under simulation scenarios from the view of two primary parameters.

7.9.1 Overall End-to-End Delay

Figures 7.10 and 7.11 illustrate the overall end-to-end delay results for the static load balancing for MMOGs based on hybrid P2P architecture and compare the results with static load balancing for MMOGs based on client-server architecture. The figures below illustrate a considerable variation of delay when using static load balancing for MMOGs hybrid P2P system with 125 and 500 players compared to the system based on client-server architecture.

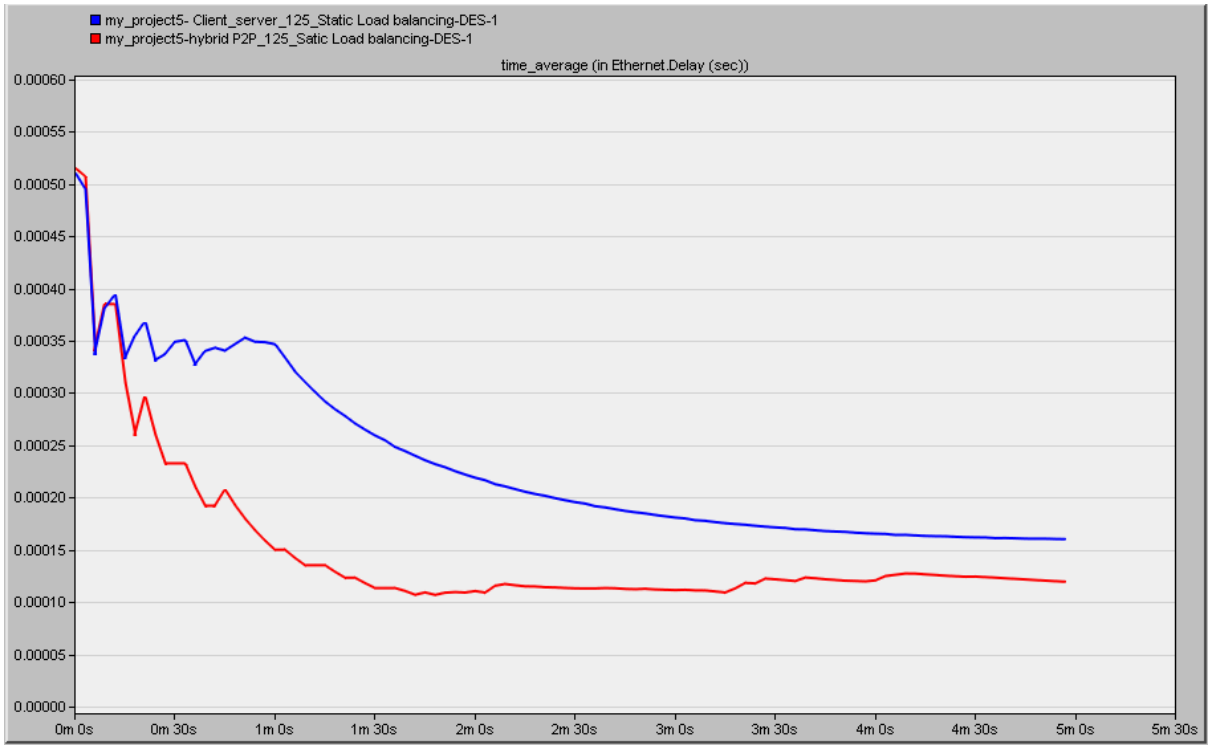


Figure 7. 10: Overall Delay for Static Load Balancing for MMOGs with 125 Peers

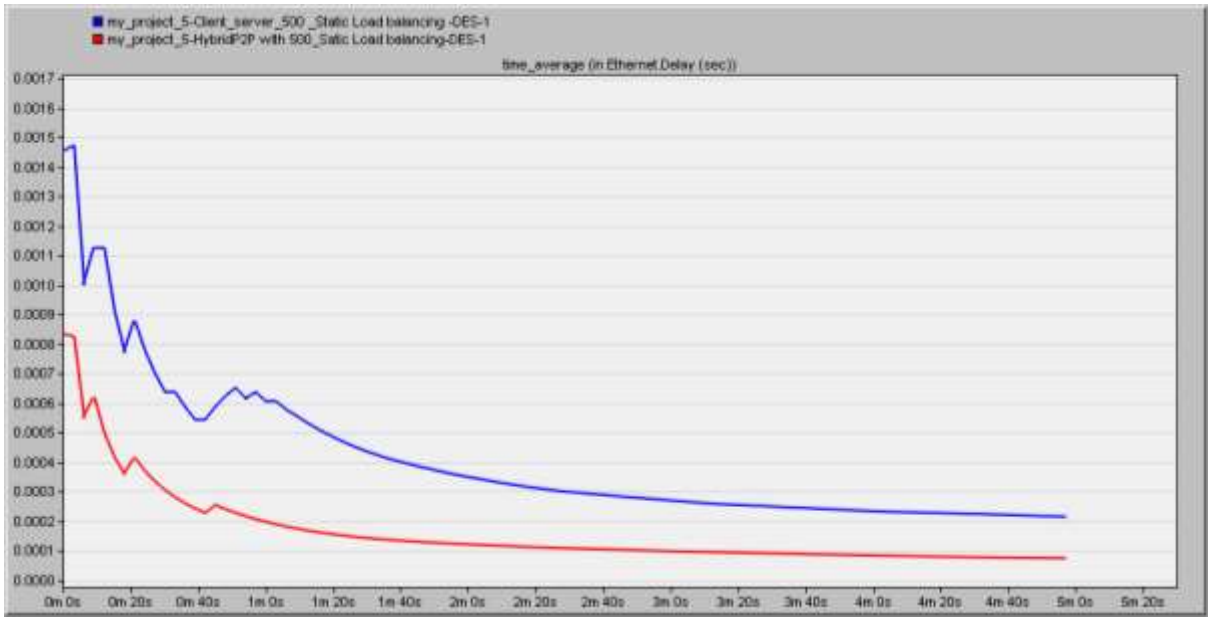


Figure 7. 11: Overall Delay for Static Load Balancing for MMOGs with 500 Peers

7.9.2 Traffic Received

Figures 7.12 and 7.13 illustrate the traffic received results for the static load balancing for MMOGs based on hybrid P2P architecture and compare the results with static load balancing for MMOGs based on client-server architecture. The figure below shows a big

variation of traffic received when using static load balancing for MMOGs based on hybrid P2P system compared to the static load balancing for MMOGs based on client-server architecture.

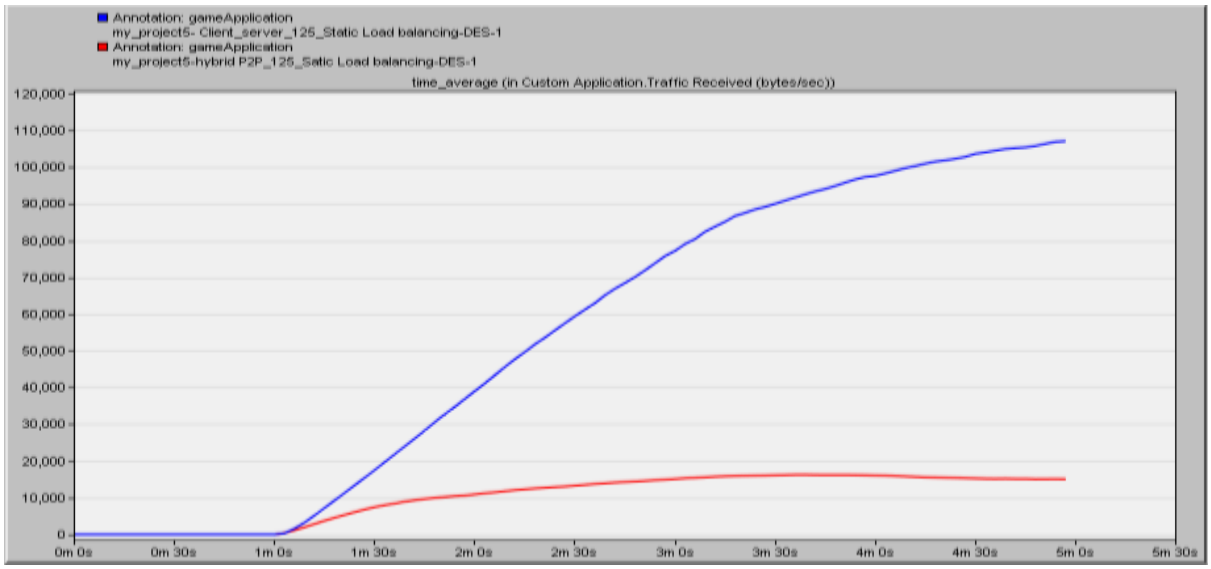


Figure 7. 12: Traffic Received for Static Load Balancing for MMOGs with 125 Peers

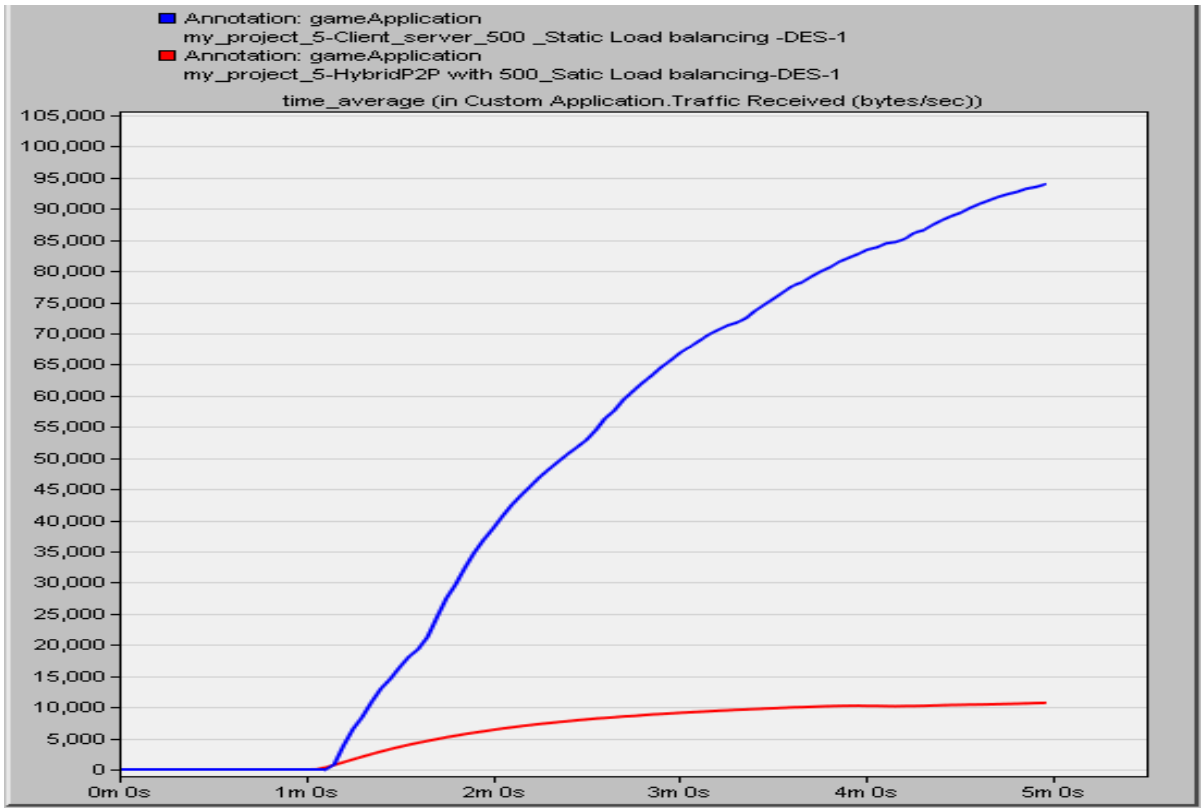


Figure 7. 13: Traffic Received for Static Load Balancing for MMOGs with 500 Peers

7.10 Simulation of Dynamic Load Balancing

In order to design the simulation for dynamic load balancing for MMOGs, we have fail all the nodes of players before the simulation. It will help us to start the game from the base. We have used the deterministic technique for generating failures and recoveries for each node (peer) in the game world space. All the nodes will start to recover according to the specific time that will be given to each node in the game world space. After the player's node recovers, it will start the game by registration in the game server. The game server is responsible for creating a profile for each player and sends this information to the load balancer in order to enable the load balancer to distribute the players among the regions inside the game world space. The load balancer is responsible for distributing the nodes (peers) among the regions in the game world space. Figures 7.14, 7.15, and 7.16 show the simulation design of dynamic load balancing for MMOGs based on hybrid P2P system with 125, 500, 1,000 peers respectively.

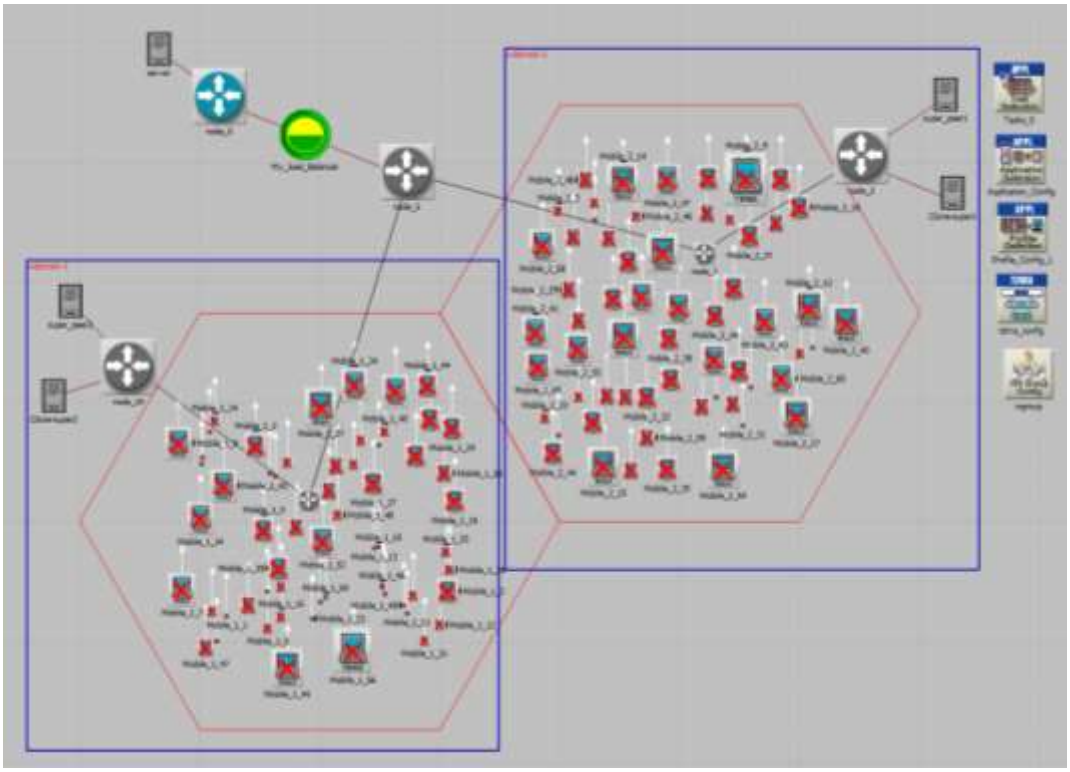


Figure 7. 14: Dynamic Load Balancing for MMOGs based on Hybrid P2P System with 125 peers

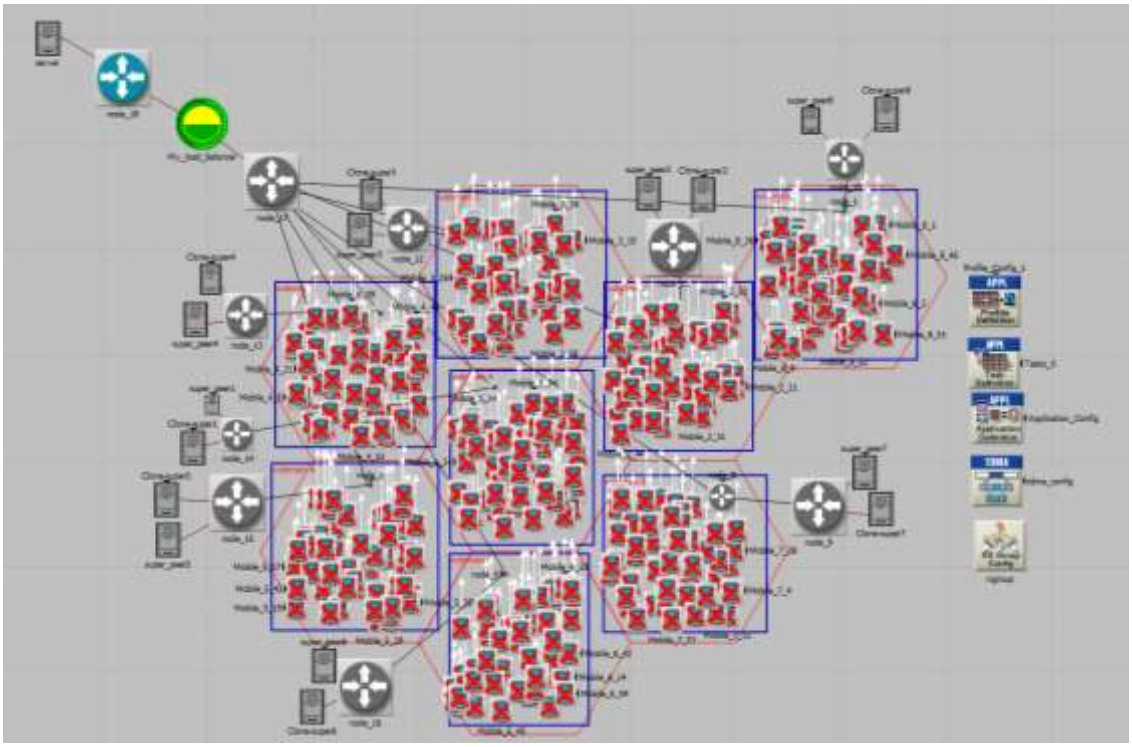


Figure 7. 15: Dynamic Load Balancing for MMOGs based on Hybrid P2P System with 500 peers

Figure 7. 16: Dynamic Load Balancing for MMOGs based on Hybrid P2P System with 1000 peers

7.11 Results of Dynamic Load Balancing

In this section, we have analysed the network performance under simulation scenarios from the view of two primary parameters.

7.11.1 Overall End-to-End Delay

Figures 7.17, 7.18, and 7.19 illustrate the overall end-to-end delay results for the dynamic load balancing for MMOGs based on hybrid P2P architecture and compare the results with static load balancing for MMOGs based on hybrid P2P architecture. The figure below illustrates a considerable variation of delay when using static load balancing for MMOGs hybrid P2P system with 125, 500, and 1,000 players compared to the static load balancing for MMOGs based on hybrid P2P architecture.

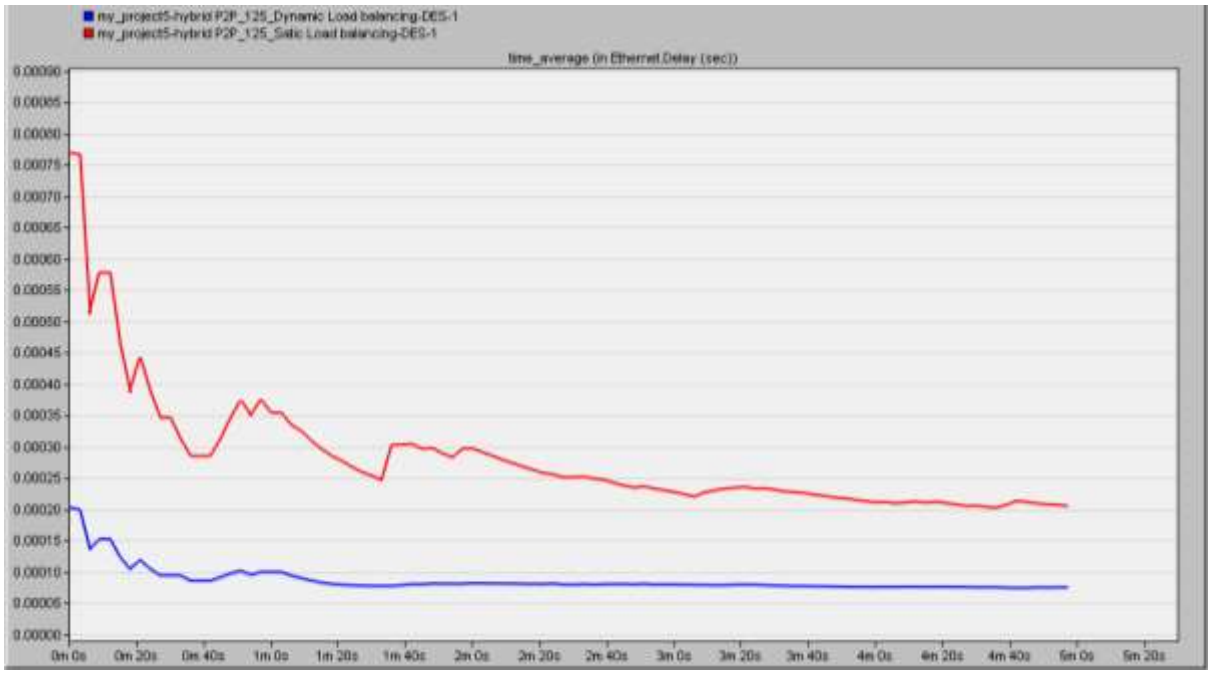


Figure 7. 17: Overall Delay for Dynamic Load Balancing for MMOGs with 125 Peers

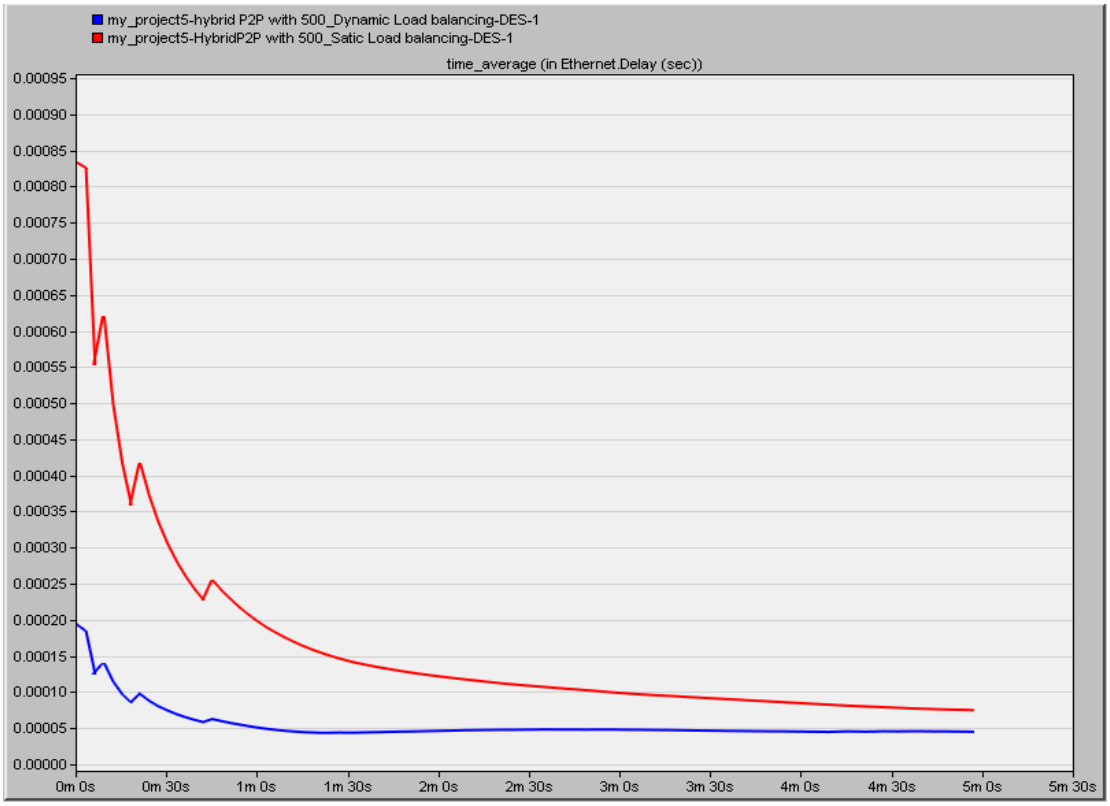


Figure 7. 18: Overall Delay for Dynamic Load Balancing for MMOGs with 500 Peers

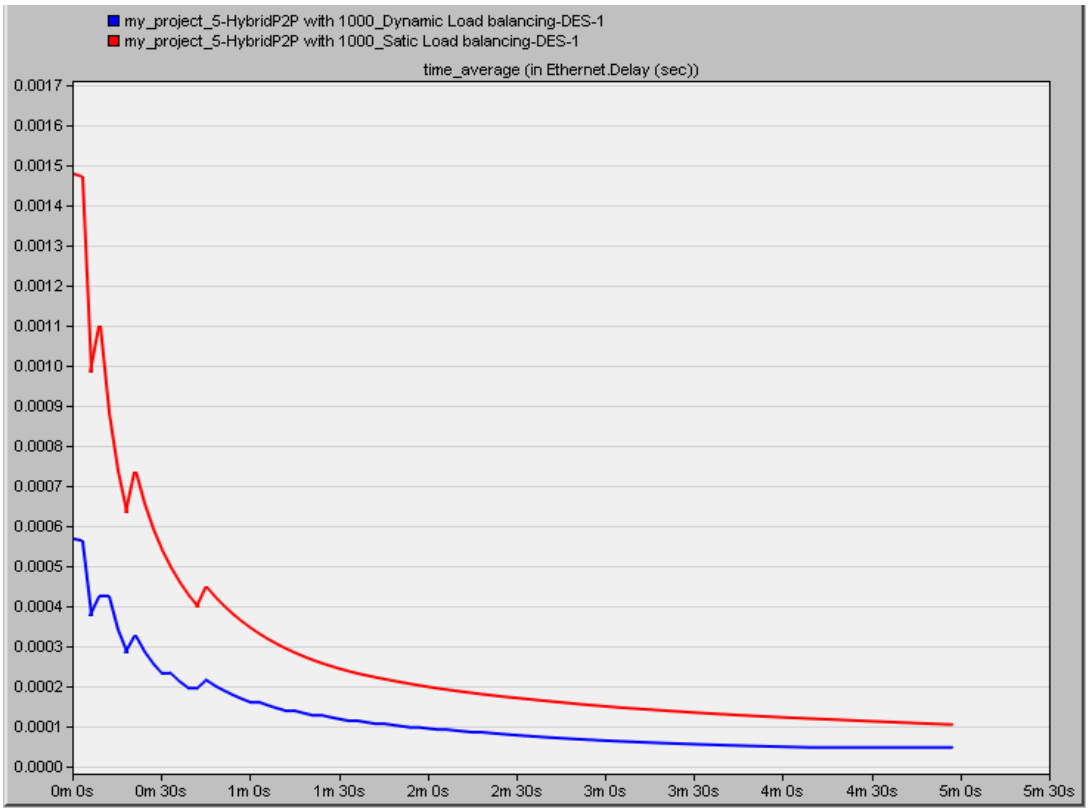


Figure 7. 19: Overall Delay for Dynamic Load Balancing for MMOGs with 1,000 Peers

7.11.2 Traffic Received

Figures 7.20, 7.21, and 7.22 illustrate the traffic received results for the dynamic load balancing for MMOGs based on hybrid P2P architecture and compare the results with static load balancing for MMOGs based on hybrid P2P architecture. The figure below shows a big variation of traffic received when using dynamic load balancing for MMOGs based on hybrid P2P system compared to the static load balancing for MMOGs based on client-server architecture.

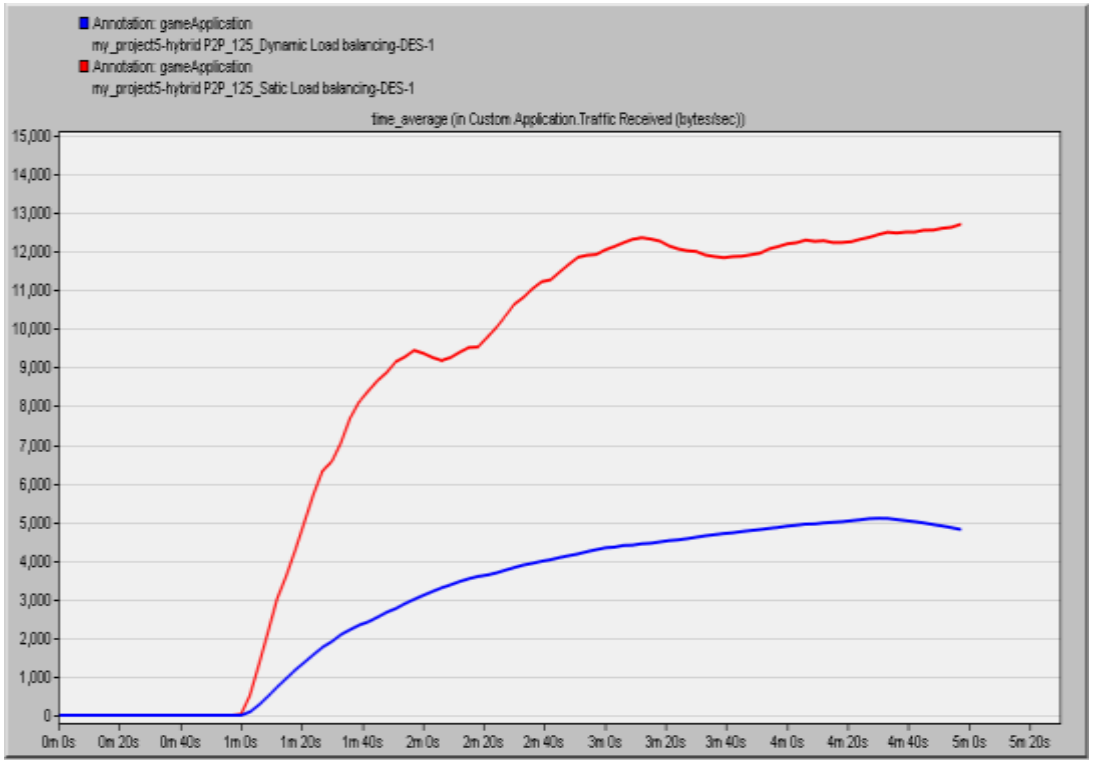


Figure 7. 20: Traffic Received for Dynamic Load Balancing for MMOGs with125 Peers

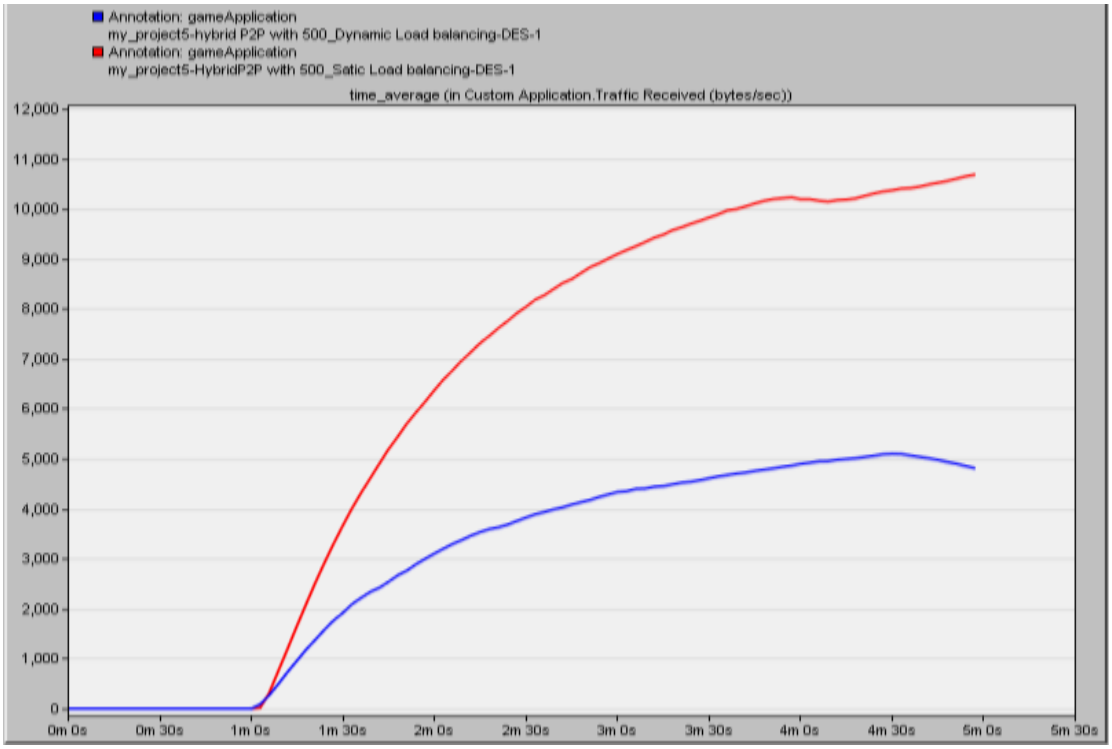


Figure 7. 21: Traffic Received for Dynamic Load Balancing for MMOGs with 500 Peers

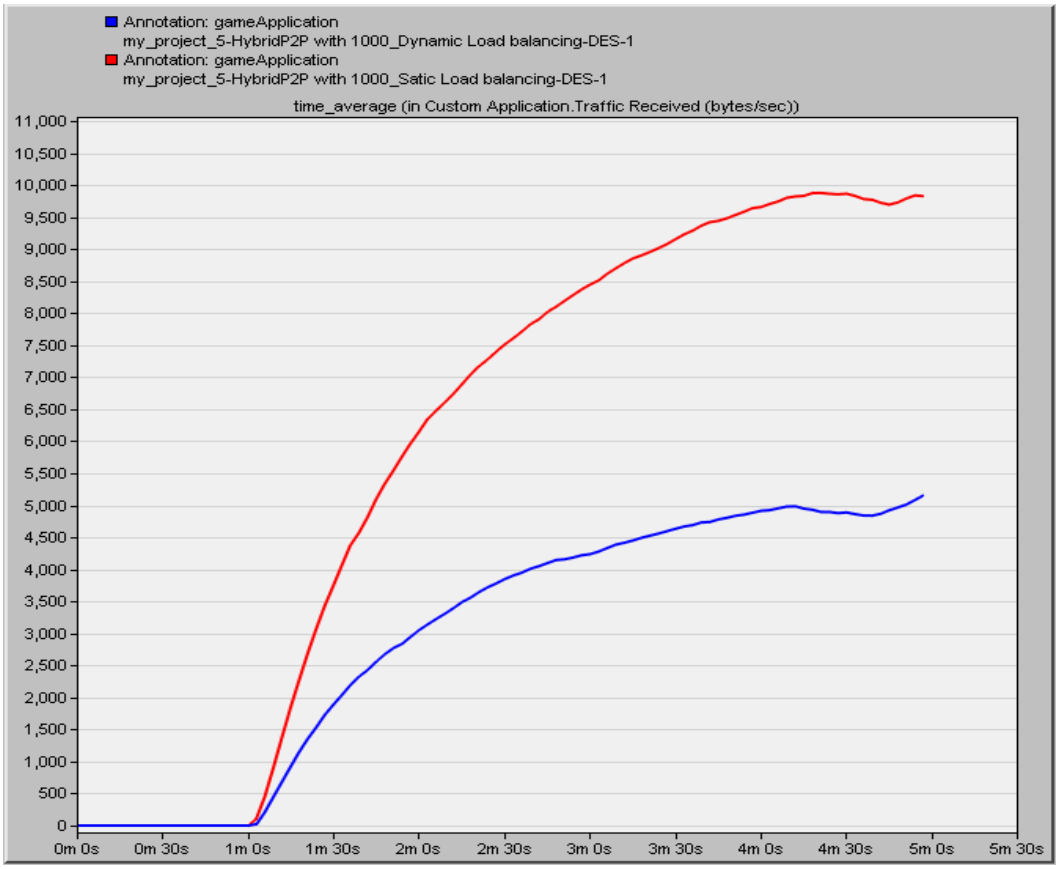


Figure 7. 22: Traffic Received for Dynamic Load Balancing for MMOGs with 1,000 Peers

7.12 Discussion

This section discusses the different results of the dynamic load balancing evaluation. Starting with the limitations that influence the results of the evaluation which are illustrated in the previous sections. Followed by the evaluation findings of the dynamic load balancing simulation.

7.12.1 Results Discussion

In general, our dynamic load balancer for MMOGs based on hybrid P2P architecture aims to reduce the costs of a traditional game server infrastructure. It provides a good level of scalability, a mechanism to deploy the players among the regions in the game world space, AoIM technique to reduce the number of receiver groups, and control of the load of each region inside the game world space compared to the load balancer for MMOGs based on client-server architecture. Also the dynamic load balancing provides an efficient mechanism to manage and control the migration of players from one region to another. The results illustrate that the use of dynamic load balancing for MMOGs based on hybrid P2P provides an easy way to manage the load of the regions and provides low delay and traffic received when compared to the load balancing for MMOGs based on client-server. Our dynamic load balancing for MMOGs has the flexibility to apply in the real life system because it has the suitable mechanism to control and manage peers joining, migrating, and leaving the game. Also, it has a robust mechanism to manage the AoIM for each player in the game world.

7.13 Conclusion

In this chapter, we have introduced the subjects and issues related to our research and explained the main contributions of the load balancing. We have also explicated the main kinds of load balancing, as well as introduced the design and simulation of dynamic load balancing for MMOGs based on both hybrid P2P and client-server system. We use OPNET Modeler 18.0 to model and simulate the new load balancing system. We have simulated two kinds of load balancing (static and dynamic load balancing) and discussed the results. We have used OPNET simulation to enable the networks construction, study of communication infrastructure, design of individual devices, and simulation of protocols and applications. The results illustrate that the static and dynamic load balancing for MMOGs hybrid Peer-to-Peer system produce low delay and low traffic received in the network topology when compared with static load balancing for MMOGs based on client-server system.

8

CONCLUSION AND FUTURE WORK

This chapter concludes the thesis and discusses possible future work. It provides a summary of the research study, reviews the research aims and objectives, discusses how they were addressed, and discusses the potential future work of the research.

8.1 The Summary of the Thesis

This research has investigated the dynamic load balancing for Massively Multiplayer Online Games (MMOGs) based on hybrid P2P architecture. The thesis started by addressing the thesis statement, aims and objectives of the thesis, and the research contributions also began in chapter one.

Research background was introduced in chapter two to provide the background of the research fields. This included explanations of concepts of MMOGs and the architectures that were used to develop MMOGs, in addition to the issues that related to the online games based on P2P architecture.

A literature review was introduced in chapter three. It contained an illustration of several researches that related to the area of study. Existing solutions for MMOGs based on Client-Server, P2P, and hybrid P2P architecture, in addition to the some solutions of load balancing for MMOGs.

Chapter four focused on the scalability of MMOGs based on hybrid P2P architecture compared with MMOGs based on client-server architecture. It discussed the challenges of MMOGs scalability for both architectures using standard OPNET Modeler applications Model. Design and simulation of MMOGs was achieved by using the applications that were provided by OPNET Modeler simulation.

Simulation of Massively Multiplayer Online Games Communication Using OPNET Custom Application was introduced in chapter five. We designed and simulated our MMOGs hybrid P2P system based on custom application and compared the simulation results with MMOGs client-server system based on custom application, in addition to, allowing the players to migrate from one region to another inside the game world space.

Chapter six introduced the Simulation of Area of Interest Management for Massively Multiplayer Online Games Using OPNET. In this chapter, we introduced the subjects and issues related to our research and explained the main contributions of the AoIM. We also explicated the area of interest management types, as well as introducing the design of both static and dynamic AoIM. We used OPNET Modeler 18.0 to model and simulate the new area of interest management system.

Chapter seven introduced the static and dynamic Load Balancing for Massively Multiplayer Online Games Using OPNET. The design and simulation of dynamic load balancing for MMOGs is explained based on both hybrid P2P and client-server system. OPNET Modeler 18.0 have been used to model and simulate the new load balancing system. We simulated two kinds of load balancing (static and dynamic load balancing) and discussed the results. We used OPNET simulation to enable the networks construction, study of communication infrastructure, design of individual devices, and simulation of protocols and applications.

Chapter eight concludes the thesis and discuss possible future work.

8.2 Meeting the objectives

The aims and objectives of the research were begun in chapter one of this thesis. The main focus was to design a novel dynamic load balancing for MMOGs based on hybrid P2P architecture. In order to achieve the main aim, various objectives were developed and satisfied during this research. The next section reviews the objectives of this research and how they were addressed.

- **To investigate the different between the MMOGs based on hybrid P2P system with MMOGs based on client-server system.**
  Chapter four illustrated the key differences between the MMOGs based on hybrid P2P system and the MMOGs based on client-server system in term of scalability, players' deployment, and organization.

- **To design a MMOGs based on both hybrid P2P architecture**

  Chapter four presented the design of primary MMOGs based on hybrid P2P architecture. The novelty of this architecture is the use of clone-super-peer concept in addition to the super-peer in order manage and control the region inside the game world. The responsibility of the super-peer are preserve connections with the normal peers, updates the game state, manage the communication between the players inside the region, Manage the joining and leaving of the players. However, the responsibility of the clone-super-peer are be as a super-peer of the region when the current super-peer crashed and suddenly leave the game in addition to the normal leaving of the super-peer. As well as managing and controlling the migration mechanism from one region to another.

- **To evaluate a MMOGs based on both hybrid P2P architecture using OPNET standard applications.**

  Chapter four also presented the evaluation of MMOGs based on hybrid P2P architecture and compared the results with the traditional client-server system. The OPNET Modeler standard applications that used to design the game are remote login, database, FTP, and HTTP. It was presented to ensure the level of scalability for MMOGs based on hybrid P2P architecture.

- **To investigate the use of custom application of OPNET Modeler to design a MMOGs based on Hybrid P2P system**.

  Chapter five presented the design and evaluation of MMOGs using OPNET Modeler custom application in order to manage and control the game in an easy way. We have used OPNET custom application to design a new game application for MMOGs based on hybrid P2P architecture and compared the results with MMOGs based on client-server architecture. In addition, we have used the possibility of player to migrate from one region to another in order to represent the game application more realistic.

- **To design and evaluate static and dynamic area of interest management (AoIM) for MMOGs.**

  Chapter six introduced the design and simulation of static and dynamic AoIM for MMOGs. In static AoIM, we have used the concept of creating a static list of receivers for each player according to its position in the game world. These lists are sending to the super-peer of the region in order to send the game updates only

to players in the list of interest. In dynamic AoIM, each player responsible to create a dynamic group of receiver and send the group to the super-peer of the region. The super-peer of the region will send the game updates just to players inside the group of receiver. We the player migrate to another region inside the game world space, the player start to create a new group of receiver. We have used OPNET Modeler TDMA and RXgroup to evaluate the dynamic AoIM.

-
- **To design a novel dynamic load balancing for MMOGs based on hybrid P2P architecture**

  Chapter seven introduce the design of novel dynamic load balancing for MMOGs based on hybrid P2P system. The ability of the system to dynamically the super-peer shift from an overloaded node to the light-load node in order to increase the response time of super-peer nodes. Our load balancing system measured the load of each super-peer in the game world space in order to find out the availability for each region.

  The measurement of the load is based on the node resources such as power of the CPU, memory available, bandwidth, disk space as well as stability and reactivity.
- **To investigate the suitability of proposed system over different numbers of players and various numbers of regions.**

  Dynamic load balancing for MMOGs based on hybrid P2P system was applied for different numbers of player, and different number of regions in order to ensure the flexibility of the system to apply in various scenarios.
- **To evaluates the performance of the proposed system.**

  Dynamic load balancing for MMOGs based on hybrid P2P system was evaluated using OPNET Modeler simulation environment. We have created a new process model and new node model of the load balancer using both process editor and node editor of OPNET. The load balancer based between the game server and the super-peers of the regions. We have compared the results with the static load balancing in order to ensure the performance when a player is joining, migrating, and leaving the game.

8.3 Limitations

The current work has the following limitations:

1- The simulation environment (OPNET) that used to evaluate the proposed system has limited number of nodes (1000 nodes).
2- The simulation network of UDP transport protocol takes extremely large time and memory. Therefore, we only simulate the scenario with 125 nodes, whilst all the other experiments use TCP.
3- The evaluation system (OPNET Modeler) does not provide the mechanism to support any data structure to store the queue of players that enables us to use them in the simulation. Also it does not provide the mechanism to split the region into two regions during the simulation.

8.4 Future Work

There are several limitations encountered in this research. These constraints represent possible future work directions for expanding the proposed system. The potential future works are listed in the following:

- The using of OPNET Modeler simulation to connect the simulation with the real world through External System Interface (esys).
- Use of other simulation environments or benchmarks from the literature to evaluate the dynamic load balancing for MMOGs and compare the results with the current research.
- The use of different numbers of players to create the region in order to ensure the scalability, responsiveness, and the organisation of the system.
- The use of other simulation environments to evaluate our proposed system in order to simulate more than 1000 nodes.
- Compare our results with other research results in order to validate our results.

References

[1] L. Fan, Solving Key Design Issues for Massively Multiplayer Online Games on Peer-to-Peer Architectures, no. May. 2009.

[2] D. Pittman and C. GauthierDickey, “A measurement study of virtual populations in massively multiplayer online games,” NetGames, pp. 25–30, 2007.

[3] “Prime World - Official Prime World Game Website. Free to play Competitive RPG from Nival.” [Online]. Available: http://en.playpw.com/. [Accessed: 22-Apr-2014].

[4] “PlanetSide 2 PC Game - Massive Combat on an Epic Scale.” [Online]. Available: http://www.planetside2.eu/. [Accessed: 22-Apr-2014].

[5] “British Legends.” [Online]. Available: http://www.british-legends.com/CMS/. [Accessed: 08-Mar-2016].

[6] “ Dungeon Crawl Stone Soup.” [Online]. Available: http://crawl.develz.org/wordpress/screenshots-old. [Accessed: 08-Mar-2016].

[7] A. Yahyavi, B. Kemme, and F. Fantasy, “Peer-to-Peer Architectures for Massively Multiplayer Online Games : A Survey,” ACM Comput. Surv., vol. 46, no. 1.

[8] C. J. Bonk and V. P. Dennen, “Massive Multiplayer Online Gaming : A Research Framework for Military Training and Education READINESS AND TRAINING , OFFICE OF THE UNDER SECRETARY,” no. March, 2005.

[9] B. Knutsson, H. Lu, W. Xu, and B. Hopkins, “Peer-to-Peer Support for Massively Multiplayer Games,” IEEE INFOCOM, 2004.

[10] M. Castro, P. Druschel, A. Kermarrec, and A. Rowstron, “S CRIBE : A large-scale and decentralized application-level multicast infrastructure,” vol. 20, no. 8, 2002.

[11] B. Hughes, J. Haggerty, J. Nothman, S. Manickam, and J. R. Curran, “A Distributed Architecture for Interactive Parse Annotation,” Proc. Australas. Lang. Technol. Work., pp. 207–214, 2005.

[12] L. Fan, “Deadline-Driven Auctions for NPC host allocation in P2P MMOGs Phil Trinder and Hamish Taylor,” vol. 4, no. 2, pp. 140–153, 2010.

[13] “Star Wars: The Old Republic Review - AusGamers.” [Online]. Available: http://www.ausgamers.com/games/star-wars-the-old-republic/review/. [Accessed: 15-Mar-2016].

[14] “Anubisath Sentinel - Hearthstone Cards.” [Online]. Available: http://www.hearthpwn.com/cards/27241-anubisath-sentinel. [Accessed: 15-Mar-2016].

[15] J. Smed, T. Kaukoranta, and H. Hakonen, “Aspects of networking in multiplayer computer games,” Turku Cent. Comput. Sci., vol. 20, no. 2, pp. 87–97, 2002.

[16] J. Chen, B. Wu, M. Delap, B. Knutsson, H. Lu, and C. Amza, “Locality aware dynamic load management for massively multiplayer games,” Proc. tenth ACM SIGPLAN Symp. Princ. Pract. parallel Program. - PPoPP ’05, p. 289, 2005.

[17] M. Suznjevic and M. Matijasevic, “Player behavior and traffic characterization for MMORPGs: a survey,” Multimed. Syst., vol. 19, no. 3, pp. 199–220, Jun. 2012.

[18] E. A. Nathan Sheldon, Eric Girard, Seth Borg, Mark Claypool, “The effect of latency on

user performance in Warcraft III," ACM, vol. May 22-23, no. USA, 2003.

[19] J. Smed, T. Kaukoranta, and H. Hakonen, "A Review on Networking and Multiplayer Computer Games Timo Kaukoranta," Turku Cent. Comput. Sci. TUCS Tech. Rep. No 454, no. Finland.

[20] I. S.-S. Board and Authorized, "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) Framework and Rules," New York, USA.

[21] C. Diot and L. Gautier, "A Distributed Architecture for Multiplayer Interactive Applications on the Internet," no. France, 2004.

[22] S. Ratnasamy, P. Francis, M. Handley, S. Shenker, and R. Karp, "A Scalable Content-Addressable Network," SIGCOMM'01, no. San Diego, California, USA.

[23] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, a. D. Joseph, and J. D. Kubiatowicz, "Tapestry: A Resilient Global-Scale Overlay for Service Deployment," IEEE J. Sel. Areas Commun., vol. 22, no. 1, pp. 41–53, Jan. 2004.

[24] H.-H. Lee and C.-H. Sun, "Load-balancing for peer-to-peer networked virtual environment," Proc. 5th ACM SIGCOMM Work. Netw. Syst. Support games - NetGames '06, p. 14, 2006.

[25] Q. Vu, M. Lupu, and B. Ooi, "Architecture of Peer-to-Peer Systems," Peer-to-Peer Comput., pp. 11–37, 2010.

[26] C. S'ebastien, Jean; Kienzle, Jorg; Verbrugge, "Comparing Interest Management Algorithms for Massively Multiplayer Games," ACm, pp. 1–12, 2006.

[27] P. Karbhari, M. Ammar, A. Dhamdhere, H. Raj, G. Riley, and E. Zegura, "Bootstrapping in Gnutella : A Measurement Study," NSF grants ANI-0240485 ANI-9973115, no. Georgia Institute of Technology, Atlanta.

[28] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet : A Distributed Anonymous Information Storage and Retrieval System."

[29] M. Izal, E. W. Biersack, P. A. Felber, A. A. H. Hamra, and L. Gautier, "Dissecting BitTorrent : Five Months in a Torrent ' s Lifetime," no. France.

[30] D. G. K. Petrozavodsk; and A. G. Linköping, "Structured Peer-to-peer Systems : Fundamentals of Hierarchical Organization , Routing , Scaling , and Security," New York Springer Sci. + Bus. Media, 2013.

[31] H. Balakrishnan, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Looking up data in P2P systems," Commun. ACM, vol. 46, no. 2, p. 43, Feb. 2003.

[32] A. Rowstron and P. Druschel, "Pastry : Scalable , decentralized object location and routing for large-scale peer-to-peer systems," ACM Int. Conf. Distrib. Syst. Platforms, no. Heidelberg, Germany, November 2001, 2001.

[33] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup protocol for internet applications," IEEE/ACM Trans. Netw., vol. 11, no. 1, pp. 17–32, Feb. 2003.

[34] P. Maymounkov and D. Mazi, "Kademlia : A Peer-to-peer Information System Based on the XOR Metric," Natl. Sci. Found. grants CCR, no. New York University.

[35] D. Malkhi, M. Naor, and D. Ratajczak, “Viceroy : A Scalable and Dynamic Emulation of the Butterfly,” PODC, no. Monterey, California, USA, 2002.

[36] T. H. H. Y. K. Iimura, “Zoned Federation of Game Servers : a Peer-to-peer Approach to Scalable Multi-player Online Games,” SIGCOMM’04Workshops, no. Portland, Oregon, USA.

[37] L. Fan and P. Trinder, “Mediator : A Design Framework for P2P MMOGs,” NetGames’07, no. Melbourne, Australia, pp. 43–48.

[38] C. Carter, “An Extensible Test bed Architecture and Topological Analysis for the Scalability of Hybrid-P2P Massively Multiplayer Online Games,” no. APRIL, 2012.

[39] A. Bharambe, J. R. Douceur, J. R. Lorch, T. Moscibroda, J. Pang, S. Seshan, and X. Zhuang, “Peer-to-Peer Games,” pp. 389–400, 2008.

[40] “The Network Simulator - ns-2.” [Online]. Available: https://www.isi.edu/nsnam/ns/. [Accessed: 07-Nov-2017].

[41] C. Mallanda, a Suri, V. Kunchakarra, S. S. Iyengar, R. Kannan, a Durresi, and S. Sastry, “Simulating wireless sensor networks with OMNeT,” Submitt. to IEEE Comput., 2005.

[42] “OPNET Modeler Simulation Environment.” [Online]. Available: http://www.riverbed.com/products/performance-management-control/opnet.html?redirect=opnet. [Accessed: 26-Sep-2014].

[43] “OPNET .” [Online]. Available: http://www.riverbed.com/products/performance-management-control/opnet.html. [Accessed: 04-Jun-2015].

[44] A. Bharambe and J. Pang, “Colyseus : A Distributed Architecture for Online Multiplayer Games.”

[45] V. M. Alecu, “Developing a client-server architecture and minimizing data transfer for a massively multiplayer online game,” Master Sci. Thesis, no. Utrecht, the Netherlands.

[46] M. Assiotis and V. Tzanov, “A Distributed Architecture for MMORPG,” pp. 1–7, 2006.

[47] M. Merabti and A. El Rhalibi, “Peer-to-Peer Architecture and Protocol for a Massively Multiplayer Online Game,” pp. 519–528, 2004.

[48] C. Gauthierdickey, D. Zappala, and V. Lo, “A Fully Distributed Architecture for Massively Multiplayer Online Games,” SIGCOMM’04Workshops, no. Portland, Oregon, p. 2004, 2004.

[49] T. Hampel, T. Bopp, and R. Hinn, “A Peer-to-Peer Architecture for Massive Multiplayer Online,” pp. 1–4, 2006.

[50] L. Chan, J. Yong, J. Bai, B. Leong, and R. Tan, “Hydra : A Massively-Multiplayer Peer-to-Peer Architecture for the Game Developer.”

[51] D. Bauer, I. Iliadis, S. Rooney, and P. Scotton, “Communication Architectures for Massive Multi-Player Games,” NetGames ’02 Proc. 1st Work. Netw. Syst. Support games, pp. 14–22, 2002.

[52] J. Chen and T. Chen, “VON: A Scalable Peer-to-Peer Network for Virtual Environments,” no. August, pp. 22–31, 2006.

[53] N. Jeppesen and R. A. J. Mønnike, “Peer-to-Peer Architecture for Massively Multiplayer

Online Games," 2011.

[54] Y. Wang, "A Fully Distributed P2P Communications Architecture for Network Virtual Environments," pp. 2–5, 2011.

[55] a Bondi, "Characteristics of Scalability and Their Impact on Performance," Proc. 2nd Int. Work. Softw. Perform., pp. 195–203, 2000.

[56] J. Jardine and D. Zappala, "A Hybrid Architecture for Massively Multiplayer Online Games," pp. 60–65, 2008.

[57] K. Kim, I. Yeom, and J. Lee, "HYMS : A Hybrid MMOG Server Architecture ∗," no. xx, pp. 1–8, 2004.

[58] A. Y. Chen, "A Hybrid Architecture for Massively Scaled Distributed Virtual Environments," UCLA Multimed. Syst. Lab.

[59] L. Yang, P. Sutinrerk, and K. Massively, "Mirrored Arbiter Architecture -- A Network Architecture for Large Scale Multiplayer Games," pp. 709–716, 2007.

[60] D. T. Ahmed and S. Shirmohammadi, "A dynamic area of interest management and collaboration model for P2P MMOGs," Proc. - 12th 2008 IEEE/ACM Int. Symp. Distrib. Simul. Real Time Appl., pp. 27–34, 2008.

[61] A. P. Yu and S. T. Vuong, "MOPAR : A Mobile Peer-to-Peer Overlay Architecture for Interest Management of Massively Multiplayer Online Games," Acm, pp. 99–104, 2005.

[62] D. T. Ahmed and S. Shirmohammadi, "An Auxiliary Area of Interest Management for Synchronization and Load Regulation in Zonal P2P MMOGs," no. October, pp. 18–19, 2008.

[63] B. Jean-S´ebastien, "Interest Management for Massively Multiplayer Games," Master thesis, no. McGill University, 2006.

[64] K. Pan, W. Cai, X. Tang, S. Zhou, and S. J. Turner, "A Hybrid Interest Management Mechanism for Peer-to-Peer Networked Virtual Environments."

[65] J. Mohorko, F. Matjaž, and K. Saša, "Advanced modelling and simulation methods for communication networks," Microw. Rev., pp. 41–46, 2008.

[66] S. Shirmohammadi, A. Diabi, and P. Lacombe, "A Peer-to-Peer Communication Architecture for Networked Games," Communication, 2005.

[67] A. Denault and J. Kienzle, "The perils of using simulations to evaluate Massively Multiplayer Online Game performance," 3rd Int. ICST Conf. Simul. Tools Tech. ICST (Institute Comput. Sci. Soc. Telecommun. Eng., 2010.

[68] S. D. Webb, W. Lauel, and S. Soh, "NGS : An Application Layer Network Game Simulator," Proceeding IE '06 Proc. 3rd Australas. Conf. Interact. Entertain., no. December, pp. 15–22, 2006.

[69] "Applications Model User Guide," in Guide Standard Models, Modeler Relase 10.0, pp. 1–36.

[70] A. Denault, C. Canas, J. Kienzle, and B. Kemme, "Triangle-based obstacle-aware load balancing for massively multiplayer games," 2011 10th Annu. Work. Netw. Syst. Support Games, pp. 1–6, Oct. 2011.

[71] C. E. B. Bezerra, J. L. D. Comba, and C. F. R. Geyer, “A Fine Granularity Load Balancing Technique for MMOG Servers Using a KD-Tree to Partition the Space,” 2009 VIII Brazilian Symp. Games Digit. Entertain., pp. 17–26, 2009.

[72] F. Lu, S. Parkin, and G. Morgan, “Load balancing for massively multiplayer online games,” Proc. 5th ACM SIGCOMM Work. Netw. Syst. Support games - NetGames ’06, p. 1, 2006.

[73] D. T. Ahmed and S. Shirmohammadi, “Uniform and Non-Uniform Zoning for Load Balancing in Virtual Environments,” 2010.

[74] S. A. Sethi and V. Y. Huatyshin, The Practical OPNET® User Guide for Computer Network Simulation. CRC Press, Taylor & Francis Group, 2013.

[75] Z. Lu and H. Yang, Unlocking the Power of OPNET Modeler, First publ. Cambridge University Press, The Edinburgh Building, Cambridge CB2 8RU, UK: The United States of America by Cambridge University Press, New York, 2012.

[76] L. Fan, “Design issues for Peer-to-Peer Massively Multiplayer Online Games,” Int. J. Adv. Media Commun., vol. 4, no. 2, pp. 108–125, 2010.

[77] M. Node and R. Modeler, “Modeling Wireless Networks,” Riverbed Man., pp. 3–51.

[78] R. Michael, R. Michael, J. Michael, R. David, T. Paul, M. J. Zyda, and D. R. Pratt, “NPSNET : A Network Software Architecture for Large Scale Virtual Environments,” vol. 3, no. 4, pp. 265–287, 1994.

[79] G. Morgan, F. Lu, and K. Storey, “Interest Management Middleware for Networked Games,” ACM, vol. 1, no. 212, pp. 57–64, 2005.

[80] S. A. Abdulazeez, A. El Rhalibi, and D. Al-jumeily, “Evaluation of Scalability and Communication in MMOGs,” 13th IEEE Annu. Consum. Commun. Netw. Conf. Eval., pp. 400–405, 2016.

[81] S. Benford, “A Spatial Model of Interaction in Large Virtual Environments,” Third Eur. Conf. Comput. Coop. Work, no. Milan, Italy, pp. 109–124, 1993.

[82] E. Buyukkaya and M. Abdallah, “Data Management in Voronoi-Based P2P Gaming,” 2008 5th IEEE Consum. Commun. Netw. Conf., pp. 1050–1053, 2008.

[83] S.-Y. Hu and G.-M. Liao, “Scalable peer-to-peer networked virtual environment,” Proc. ACM SIGCOMM 2004 Work. NetGames ’04 Netw. Syst. Support games - SIGCOMM 2004 Work., p. 129, 2004.

[84] I. Kazem, D. T. Ahmed, and S. Shirmohammadi, “A Visibility-Driven Approach to Managing Interest in Distributed Simulations with Dynamic Load Balancing,” 11th IEEE Int. Symp. Distrib. Simul. Real-Time Appl., pp. 31–38, Oct. 2007.

[85] T. Hampel, T. Bopp, and R. Hinn, “A peer-to-peer architecture for massive multiplayer online games,” Proc. 5th ACM SIGCOMM Work. Netw. Syst. Support games - NetGames ’06, p. 48–es, 2006.

[86] C. D. TOth, “Binary Space Partitions : Recent Developments,” Comb. Comput. Geom., vol. 52, pp. 529–556, 2005.

[87] A. E. L. Rhalibi and M. Merabti, “Agents-Based Modeling for a Peer-to-Peer MMOG Architecture School of Computing and Mathematical Sciences , Liverpool John,” vol.

3, no. 2, 2005.

[88] S. A. Abdulazeez, “Survey of Solutions for Peer-to-Peer MMOGs,” pp. 1106–1110, 2015.

[89] J. C. S. Lui, M. F. Chan, and S. Member, “An Efficient Partitioning Algorithm for Distributed Virtual Environment Systems,” vol. 13, no. 3, pp. 193–211, 2002.

[90] Y. Deng and R. W. H. Lau, “Dynamic Load Balancing in Distributed Virtual Environments using Heat Diffusion,” vol. 0, no. 0, pp. 1–20, 2013.

[91] H. Jiang, A. Iyengar, E. Nahum, W. Segmuller, A. Tantawi, and C. P. Wright, “Design , Implementation , and Performance of A Load Balancer for SIP Server Clusters.”

[92] S. Zhou, “A Dynamic Load Sharing Algorithm for Massively Multiplayer Online Games.”

[93] S. Abdulazeez, A. El Rhalibi, M. Merabiti, and A.-J. Dhiya, “Survey of Solutions for Peer-to-Peer MMOGs,” ICNC 2015, no. California, USA.

[94] R. Chertov and S. Fahmy, “Optimistic load balancing in a distributed virtual environment,” Proc. 2006 Int. Work. Netw. Oper. Syst. Support Digit. audio video - NOSSDAV ’06, p. 1, 2006.

[95] C. Link, Q. H. Vu, C. Ooi, M. Rinard, and K. Tan, “Histogram-Based Global Load Balancing in Structured Peer-to-Peer Systems,” 2014.

[96] R. Naaz, Sameena; Alam, Afshar; Biswas, “Load Balancing Algorithms for Peer to Peer and Client Server Distributed Environments,” vol. 47, no. 8, pp. 17–21, 2012.

[97] J. Lim, J. Chung, J. Kim, and K. Shim, “A Dynamic Load Balancing for Massive Multiplayer Online Game Server.”

[98] S. Rajani and N. Garg, “A Clustered Approach for Load Balancing in Distributed Systems,” vol. 2, no. 1, pp. 1–6, 2015.

[99] C. Science and M. Studies, “A comparative study of static and dynamic Load Balancing Algorithms,” pp. 386–392, 2014.

[100] D. Wadhwa and N. Kumar, “Performance Analysis of Load Balancing Algorithms in,” vol. 4, no. 1, pp. 59–66, 2014.

[101] N. K. Mishra, “Load Balancing Techniques : Need , Objectives and Major Challenges in Cloud Computing- A Systematic Review,” vol. 131, no. 18, pp. 11–19, 2015.

[102] S. Malik, “Dynamic Load Balancing in a Network of Workstations,” 95.515F Res. Rep.

[103] S. Sharma, S. Singh, and M. Sharma, “Performance Analysis of Load Balancing Algorithms,” World Acad. Sci. Eng. Technol., vol. 38, pp. 269–272, 2008.

[104] A. M. Alakeel, “A Guide to Dynamic Load Balancing in Distributed Computer Systems,” vol. 10, no. 6, pp. 153–160, 2010.

[105] P. B. Soundarabai, S. R. A, R. K. Sahai, J. Thriveni, K. R. Venugopal, and L. M. Patnaik, “Comparative Study on Load Balancing Techniques in Distributed Systems,” vol. 6, no. 1, pp. 53–60, 2012.

[106] P. M. Operation, “Process Domain,” OPNET Model. 18.0, vol. chapter 5, pp. 1–116.

[107] “Network Domain,” OPNET Model. 18.0, pp. 291–323.