# DEVELOPMENT AND PERFORMANCE TESTING OF ONLINE TEXT-BASED STRATEGY GAMES

By

Mohd Khairul Zaime b. Razali

Final Draft submitted in partial fulfilment of
the requirements for the
Bachelor of Technology (Hons)
(Information and Communication Technology)

JAN 2008

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

# CERTIFICATION OF APPROVAL

**DEVELOPMENT AND PERFORMANCE TESTING OF ONLINE TEXT-BASED STRATEGY GAMES**

By

Mohd Khairul Zaime b. Razali

Final Draft submitted in partial fulfilment of
the requirements for the
Bachelor of Technology (Hons)
(Information and Communication Technology)

JAN 2008

Approved by,

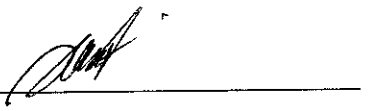
_____

(MRS. NAZLEENI SAMIHA HARON)

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

# CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in references and acknowledgements, and that original work contained herein have not been undertaken or done by unspecific sources or persons.



Mohd Khairul Zaime b. Razali

# ABSTRACT

This final draft is to show the results and achievements of the project since the beginning until it is completed. The development of an online text-based game and the load tests ran on the game to compare the performance of the game against the standard acceptable level of performance to prove that text-based games can and should be able to provide acceptable level of performance so that everyone can experience it and have an enjoyable time playing it. This project shows that the online text-based game developed during the project are able to provide performances that are above the standard acceptable level of performance after enduring a load test. The focus of the project is toward developing an online text-based strategy game using content development and testing and comparing the performances of the game in terms of throughput, scalability, response time and page loading time. The objective is to compare the performances of the developed online text-based game against the standard acceptable level of performance using the data obtained from conducting load tests on the text-based game. In the project that consists of two phases, I used content development to develop an online text-based game that is used in the testing phase as the first phase and then conducting the load test on the game and compare the results against the standard acceptable level of performance as the second phase. The comparison are done by using tools to measure and analyze several parameters of the game's performance which is throughput, scalability, response time and page loading time to determine the performance of the game against the standard acceptable level of performance. The methodology used is Agile software development model which is RUP and AUP. This project shows that online text-based games are supposed to be lightweight in nature to enable everyone to play the game to its full potential and enjoying it. I hope that this project will help people to understand how an online text-based game should perform and also to provide an online text-based game that meets the standard acceptable level of performance.

# ACKNOWLEDGEMENT

First and foremost, I would like to thank Allah the Almighty for His blessings that made all things possible while doing this project. I would like to express my deepest gratitude to my parents for their love and support. To my supervisor, Mrs. Nazleeni Samiha Haron, thank you so much for the guidance and support that she gave throughout the project. Without her advices and helps, this project may not be able to be completed within the given timeline. I also would like to thank other lecturers for their suggestions to improve the project.

I would also like to thank the Quantum Star developer for providing the template and coding for me to improve in developing the text-based strategy game.

Last but not least, I would like to thank all my friends for the help and support that they have given me throughout the period of this project. They are the one who tested my system and with their feedback and suggestions I manage to achieve what I have now.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# CHAPTER 1
# INTRODUCTION

## 1.1 BACKGROUND OF STUDY

Online text-based games are games that use text characters instead of bitmapped or vector graphics and are integrated inside a web page and users need to access it using a web browser in order to play it. The games can be of any genre such as role-playing, strategy, turn-based and simulation. Most of these games are usually free of charge but there are some games that charge its users a fee to play. These pay-to-play games usually offer better user support comparing to the free ones. Users from all over the world are able to play these games and interact with each other inside the game. This can help people to communicate and socialize as well as filling up their free time.

In this project, I have developed an online text-based strategy game and conducted load tests on the game to obtain the data on the game's performances. The data are then compared against the standard acceptable level of performance to see whether the game meets the standard. The platform that I used is HTML. HTML is the predominant markup language used for web pages and is widely used nowadays.

I used the concept of content development to develop the online text-based game in HTML and PHP and then compare the performances of the game using tools that can monitor and analyze the game's throughput, scalability, load time and page loading time.

This project is beneficial in the way that I have proved that online text-based game can be lightweight in terms of load so that everyone can enjoy the game. I also have developed an online text-based game that meets the standard acceptable level of performance and is fun to play.

## 1.2 PROBLEM STATEMENT

Nowadays, there are many online text-based games in the Internet. Most of these games are using the HTTP/HTML web approach. In these games, the performance is vital as good performance means good user experience for the players to keep them playing.

Even though these games are supposed to be lightweight in terms of performance since the interface did not involve heavy use of graphic which makes text-based games ideal for everyone to experience a smooth running game, but some of the text-based games did not offer performances that is acceptable to the players. Most of the time, players will have to wait for a page that they requested, for example when after they updated their game profile, to be loaded up as data from the server is displayed. They will have to wait for a blank screen to be filled with data loaded from the server before they can interact with the page. This is not a good user experience as they might feel bored when having this problem.

Also, these games sometimes shows significant reduction in performance and sometimes gives an error to the player especially when there are a lot of players are online simultaneously. This problem should be avoided as these games should be able to cater to at least a moderate amount of simultaneous players.

The response time that some of these games take to process and send back a reply to the user are also below the accepted level of performance. This will cause delay on the client side.

The most affected group of players is those that are playing on a low network connection as these problems will make them experience bad user experience and they cannot fully enjoy the game although it is only a text based game.

By developing a text-based game and then running load tests on the game to see whether the game is able to give performances that is above the accepted level of performances to show that text-based games can really be played by everyone to the fullest without any exceptions.

## 1.3 OBJECTIVES

Below are the objectives of this project:

1. To develop an online text-based strategy game.

2. To run load tests on the developed online text-based game to obtain the data on the game's performance in four parameters which is throughput, scalability, response time and page loading time.

3. To compare the data from the load tests against the standard acceptable level of performance to see whether the developed online text-based game meets the standard acceptable level of performance.

## 1.4 SCOPE OF STUDY

My scope of study is focusing on developing an online text-based strategy game that requires users to create an account first before playing. The strategy game will be built in HTML and PHP.

Then, after the game is developed, I have conducted a series of load tests to measure the performances of the game. The load tests were conducted using several tools that can monitor and analyze the game's performances. From then, the results of the load tests were compared against the standard acceptable level of performance and from the comparison, I have showed that the online text-based game has meet the standard acceptable level of performance.

# CHAPTER 2
# LITERATURE REVIEW

## 2.1 HTML

HTML stands for Hyper Text Markup Language. Markup language provides a way to combine a text and extra information about it. The extra information, including structure, layout, or other information, is expressed using *markup*, which is typically intermingled with the primary text. It is the predominant markup language used for web pages. It provides a means to describe the structure of text-based information in a document like denoting certain text as headings, paragraphs, lists, and so on and also to supplement that text with *interactive forms*, embedded *images*, and other objects. HTML is written in the form of labels (known as tags), surrounded by angle brackets. HTML can also describe, to some degree, the appearance and semantics of a document, and can include embedded scripting language code which can affect the behavior of web browsers and other HTML processors.

Since its introduction on July, 1993, HTML has been used extensively to develop web pages and contents although it lacks some of the features found in earlier hypertext systems, such as typed links, transclusion, source tracking, fat links, and more. Even some hypertext features that were in early versions of HTML have been ignored by most popular web browsers until recently, such as the link element and in-browser Web page editing.

HTML has undergone many changes since 1993 until the final version of HTML which is HTML 4.01 was released on December 24, 1999.

## 2.2 ONLINE GAMES

Online games are games that are integrated inside a web browser and then user will have to load the web page in order to play the game. They are distinct from normal video and computer games in that they do not require any client side software to be installed. Games like this usually need the user to create a personal account that they will have to use to play the game by accessing the web page.

Online games are normally hosted on a web server that can handle a lot of traffic since usually many users will login at the same time. The database used to create online games must be able to interact quickly with server in order to provide users with good response time when users requested something.

Some examples of online web-based games are Utopia, Online Football Manager and Neopets.

## 2.3 TEXT-BASED GAMES

Text-based games are games that use text characters instead of bitmapped or vector graphics. Text games are typically easier to write and require less processing power than graphical games, and thus were more common from 1970 to 1990.

Text-based games predate graphical online games by several years, and can be attributed to the first attempts to bring multiplayer gaming to the internet.

An online text-based game is a game that is played online using solely text-based interface. Users still use their mouse to navigate and select elements of the game that they want to interact with, but all the input and display will mostly be in text with the exception of light use of static graphic.

Examples of online text-based games are Utopia, ezRPG and Earth: 2025.

## 2.4 STRATEGY GAMES

Strategy games are games that focus on gameplay requiring careful and skillful thinking and planning in order to achieve victory. In most strategy games, the player is given a full view of the game world, indirectly controlling the units under his command.

The origin of strategy games is related to their close counterpart, board games. Strategy games instantiated on computers generally take one of four archetypal forms, depending on whether the game is turn-based or real-time and whether the game's focus is upon military strategy or tactics.

Examples of strategy games are Sid Meier's Civilization series, Gunbound, Command & Conquer series and Warhammer: Dark Omen.

## 2.5 TURN-BASED STRATEGY GAMES

A turn-based strategy game is a strategy game (usually some type of wargame, especially a strategic-level wargame) that is turn-based. The phrase turn-based is used to distinguish such games from real-time strategy games (which works as in real time situation).

In this type of game, each player will be given a number of turns for them to spend in order to play the game. Most of the players action will use up a certain amount of their turns so players will need to calculate and plan the use of their turns carefully so that they will received the maximum benefits and avoid wasting their turns. The amount of useable turns of each player will increase either as time passes or by meeting certain requirement of the game if there is any.

Some examples of turn-based strategy games are Earth: 2025, Civilization and Heroes of Might and Magic.

## 2.6 LOAD TESTING

Load testing is the process of creating demand on a system or device and measuring its response.

Load testing generally refers to the practice of modeling the expected usage of a software program by simulating multiple users accessing the program's services concurrently. As such, this testing is most relevant for multi-user systems, often one built using a client/server model, such as web servers. However, other types of software systems can be load-tested also. For example, a word processor or graphics editor can be forced to read an extremely large document; or a financial package can be forced to generate a report based on several years' worth of data. The most accurate load testing occurs with actual, rather than theoretical, results.

Load testing helps to identify the maximum operating capacity of an application as well as any bottlenecks that might interfere with its operating at capacity. The basic approach to performing load testing on a Web application is:

1. Identify the performance-critical scenarios.
2. Identify the workload profile for distributing the entire load among the key scenarios.
3. Identify the metrics that you want to collect in order to verify them against your performance objectives.
4. Design tests to simulate the load.
5. Use tools to implement the load according to the designed tests, and capture the metrics.
6. Analyze the metric data captured during the tests.

There are many reasons for load-testing a Web application. The most basic type of load testing is used to determine the Web application's behavior under both normal and anticipated peak load conditions. To begin load testing, it is recommended to start with a small number of virtual users and then incrementally increase the load from normal to peak. Developers can then observe how the application performs during this gradually increasing load condition. Eventually, it

will cross a threshold limit for the performance objectives. For example, continue to increase the load until the server processor utilization reaches 75 percent, or when end-user response times exceed 8 seconds.

The following are useful inputs for load-testing a Web application:

- Performance-critical usage scenarios
- Workload models
- Performance acceptance criteria
- Performance metrics associated with the acceptance criteria
- Interview feedback from the designer or developer of the Web application
- Interview feedback from end users of the application
- Interview feedback from the operations personnel who will maintain and manage the application

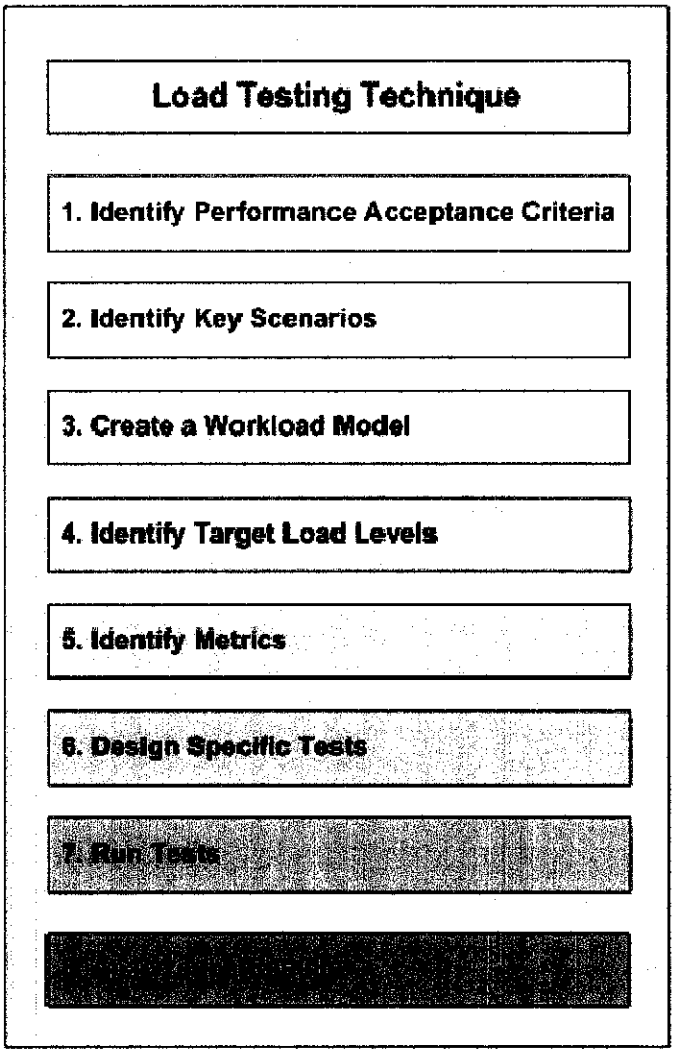
The main outcomes that load testing helps to accomplish are:

- Updated test plans and test designs for load and performance testing
- Various performance measures such as throughput, response time, and resource utilization
- Potential bottlenecks that need to be analyzed in the white-box testing phase
- The behavior of the application at various load levels

The following steps are involved in load-testing a Web application:

1. Step 1 - Identify performance acceptance criteria
2. Step 2 - Identify key scenarios
3. Step 3 - Create a workload model
4. Step 4 - Identify the target load levels
5. Step 5 - Identify metrics
6. Step 6 - Design specific tests
7. Step 7 - Run tests
8. Step 8 - Analyze the results

Figure 1: Load Testing Steps




Step 1 - Identify Performance Acceptance Criteria

Identifying performance acceptance criteria is most valuable when initiated early in the application's development life cycle. It is frequently valuable to record the acceptance criteria for the application and store them in a place and format that is available for review and comment. Criteria are typically determined by balancing the business, industry, technology, competitive, and user requirements.

Test objectives frequently include the following:

- Response time.
- Throughput.
- Resource utilization.

- Maximum user load.
- Business related metrics.

Step 2 - Identify Key Scenarios

Scenarios are anticipated user paths that generally incorporate multiple application activities. Key scenarios are those that have specific performance goals, those considered to be high-risk, those that are most commonly used, or those with a significant performance impact. The basic steps for identifying key scenarios are.

1. Identify all the scenarios for a Web application.
2. Identify the activities involved in each of the scenarios.
3. Identify the scenarios that are most commonly executed or most resource-intensive; these will be the key scenarios used for load testing.

Once they have been identified, these key scenarios will be used to create workload profiles and to design load tests.

Step 3 - Create a Workload Model

When defining workload distribution, consider the following key points for determining the characteristics for user scenarios:

- A user scenario is defined as a navigational path, including intermediate steps or activities, taken by the user to complete a task. This can also be thought of as a user session.
- A user will typically pause between pages during a session. This is known as user delay or think time.
- A session will have an average duration when viewed across multiple users. It is important to account for this when defining the load levels that will translate into concurrent usage, overlapping users, or user sessions per unit of time.
- Not all scenarios can be performed by a new user, a returning user, or either; know who you expect your primary users to be and test accordingly.

Step 4 - Identify Target Load Levels

Identify the load levels to be applied to the workload distribution(s) identified during the previous step. The purpose of identifying target load levels is to ensure that the tests can be used to predict or compare a variety of production load conditions.

Step 5 - Identify Metrics

There is a virtually unlimited number of metrics that can be collected during a performance test execution. However, collecting too many metrics can make analysis unwieldy as well as negatively impact the application's actual performance. For these reasons, it is important to identify the metrics that are most relevant to the performance objectives and those that are anticipated to help identify bottlenecks. Only well-selected metrics that are analyzed correctly and contextually provide information of value.

To evaluate the performance of your application in more detail and to identify potential bottlenecks, it is frequently useful to monitor metrics in the following categories:

- Network-specific metrics. This set of metrics provides information about the overall health and efficiency of the network, including routers, switches, and gateways.
- System-related metrics. This set of metrics helps to identify the resource utilization on the server. The resources being utilized are processor, memory, disk I/O, and network I/O.
- Platform-specific metrics. Platform-specific metrics are related to software that is used to host the application, such as the Microsoft .NET Framework common language runtime (CLR) and ASP.NET-related metrics.
- Application-specific metrics. These include custom performance counters inserted in the application code to monitor application health and identify performance issues.
- Service-level metrics. These metrics can help to measure overall application throughput and latency, or they might be tied to specific business scenarios.

Step 6 - Design Specific Tests

Using the scenarios, key metrics, and workload analysis, developers can now design specific tests to be conducted. Each test will generally have a different purpose, collect different data, include different scenarios, and have different target load levels. The key is to design tests that will help collect the information it needs in order to understand, evaluate, or tune the application.

Step 7 - Run Tests

Poor load simulations can render all of the work in the previous activities useless. To understand the data collected from a test execution, the load simulation must reflect the test design. When the simulation does not reflect the test design, the results are prone to misinterpretation. Consider the following steps when preparing to simulate load:

1. Configure the test environment in such a way that it mirrors the production environment as closely as possible, noting and accounting for all differences between the two.
2. Ensure that performance counters relevant for identified metrics and resource utilization are being measured and are not interfering with the accuracy of the simulation.
3. Use appropriate load-generation tools to create a load with the characteristics specified in the test design.
4. Using the load-generation tool(s), execute tests by first building up to the target load specified in the test design, in order to validate the correctness of the simulation. Some things to consider during test execution include:

    - Begin load testing with a small number of users distributed against the user profile, and then incrementally increase the load. It is important to allow time for the system to stabilize between increases in load while evaluating the correctness of the simulation.
    - Consider continuing to increase the load and record the behavior until the threshold for the resources identified in the performance

objectives are reached, even if that load is beyond the target load specified in the test design. Information about when the system crosses identified thresholds is just as important as the value of the metrics at the target load of the test.

- Similarly, it is frequently valuable to continue to increase the number of users until reaching the service-level limits beyond which the throughput, response time, and resource utilization will be violated.

Step 8 - Analyze the Results

Developers can analyze the test results to find performance bottlenecks between each test run or after all testing has been completed. Analyzing the results correctly requires training and experience with graphing correlated response time and system data.

The following are the steps for analyzing the data:

1. Analyze the captured data and compare the results against the metric's accepted level to determine whether the performance of the application being tested shows a trend toward or away from the performance objectives.
2. Analyze the measured metrics to diagnose potential bottlenecks. Based on the analysis, if required, capture additional metrics in subsequent test cycles. For example, suppose that during the first iteration of load tests, the process shows a marked increase in memory consumption, indicating a possible memory leak. In the subsequent iterations, additional memory counters related to generations can be captured to study the memory allocation pattern for the application.

# CHAPTER 3
# METHODOLOGY

## 3.1 PROCEDURES

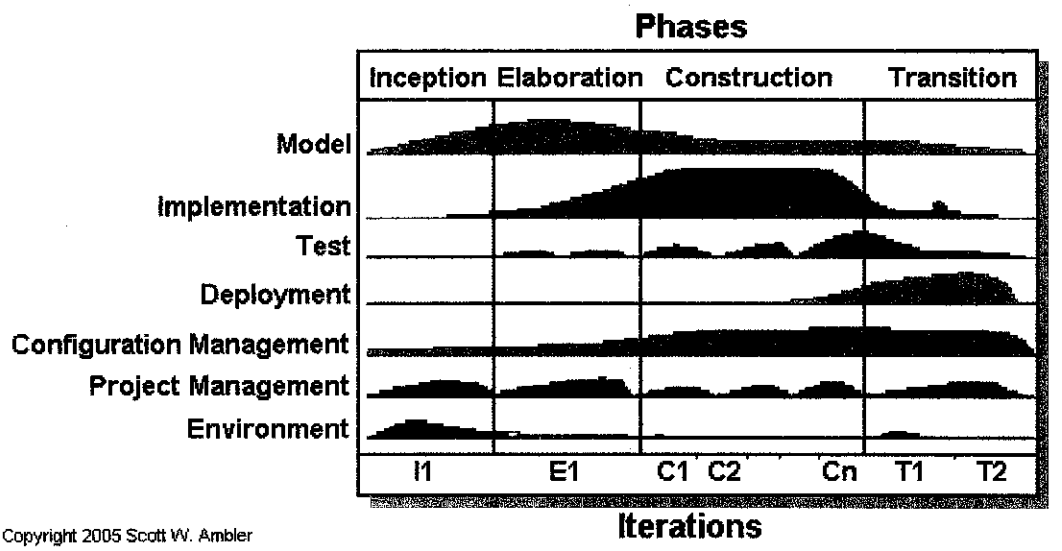
In this project, I am using the methodology called agile unified process (AUP) model. This model is using the iterative development framework and is a simplified version of the Rational Unified Process (RUP). It is designed so that developers are able to come out with a software prototype in each iteration and then improved on the feedback gathered on the prototype in the next iteration. The processes of releasing the prototypes are still using the basic design task: planning, requirements analysis, design, coding, testing, and documentation.

## 3.2 AGILE UNIFIED PROCESS

This model is suitable for the project because in the development of the game, prototypes of the game can be develop quickly and tested for its functionality. If there is any problem, it can be fixed before moving on to the next prototype with other functionalities.

The model is still using the RUP's disciplines which are Inception, Elaboration, Construction and Transition but at the same time still releasing prototypes in every iteration.

Figure 2: AUP Lifecycle



### 3.2.1 Inception:

In this phase, the scope of the project is prepared. If the project does not pass the Lifecycle Objective Milestone, it can be either cancelled or redesigned. The possible deliverables from this phase are as shown below:

- An exploration of clients requirements
- Initial risk assessment
- A project plan

### 3.2.2 Elaboration:

The problem domain analysis is done here. The architecture of the project starts to grow and gets its basic form. The project needs more assessment and a lot of project plan must be developed at this stage. There are two main UML models required in Elaboration phase which are the Use Case diagram and the Class diagram. The developer can start to develop the prototype. This phase must pass the Lifecycle Architecture Milestone. If the project fails, it can be cancelled or redesigned. It is very costly to do in the next phase.

### 3.2.3 Construction:

The developer starts to develop the intended project. In this phase, the main focus is to build the components and other features of the system being designed earlier. Here is where the programmer begins to do the coding. Several construction iterations may be developed in order to create a very good demonstrable prototype. The Construction phase produces the first external release of the software. This phase also has a milestone called Initial Operational Capability Milestone.

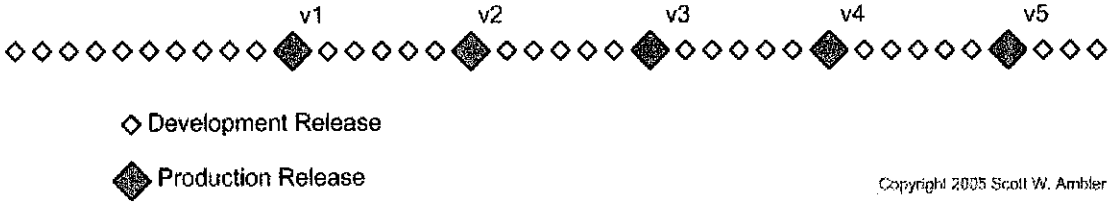
### 3.2.4 Transition:

In the Transition phase, the product is introduced to the end user. The quality of the product is checked to ensure it follows the quality level set in the Inception phase. If it does not meet the level or the standards of the end user's requirement, the entire cycle in this phase will be done again. After reached the Product Release Milestone the product is ready to use and ends its development process. Below are the main activities:

- Train the end user how to use the system

- Do a beta testing on the system
- Validate whether the product meets the user's expectations.

Figure 3: Incremental release over time




This model is good for this project, as in the first phase of the project, it involves creating an online game which is web based that can be released in a short period of time. The main characteristics of AUP that is beneficial to this project are:

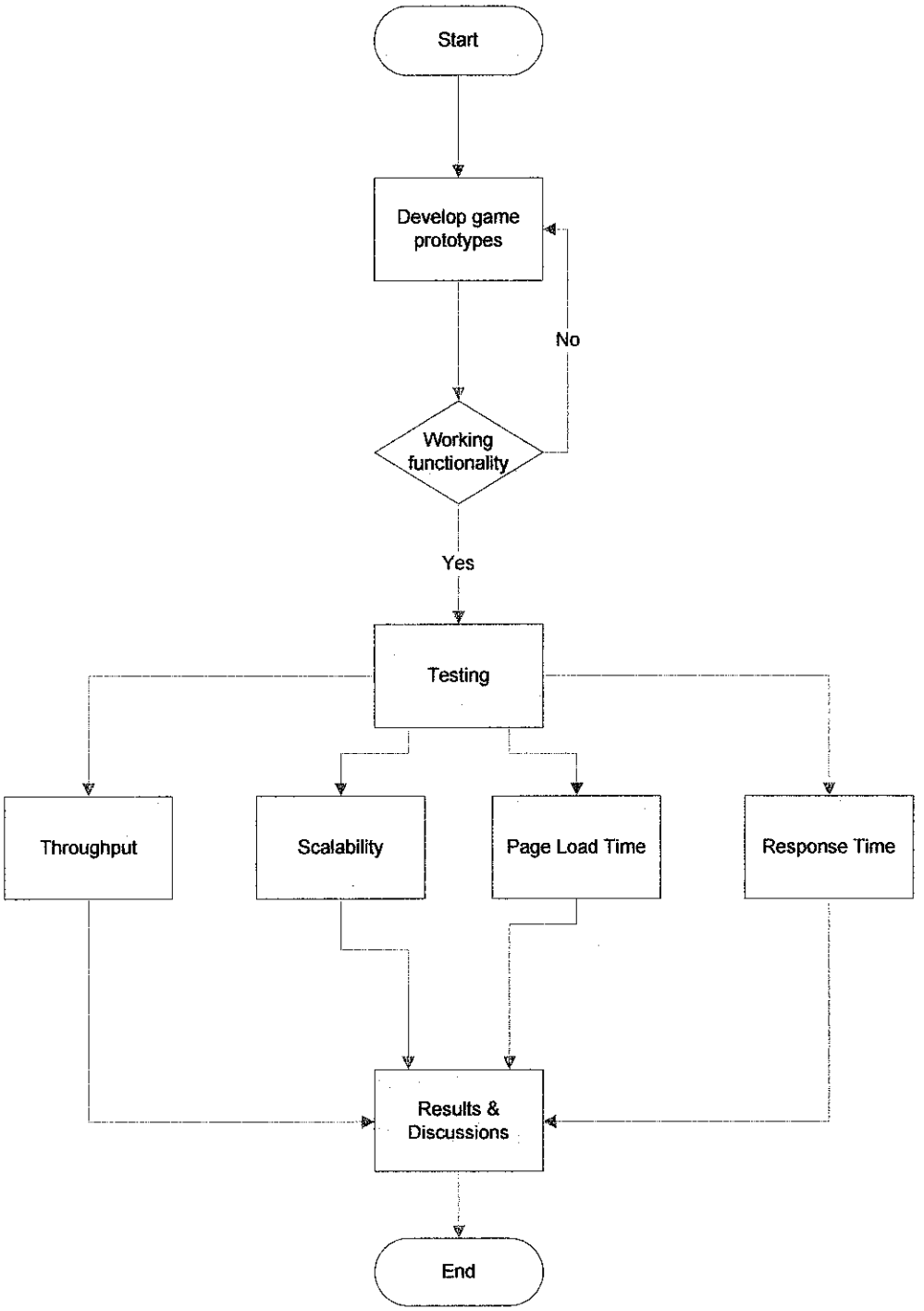
- Working software is delivered frequently (weeks rather than months)
- Working software is the principal measure of progress
- Even late changes in requirements are welcomed
- Continuous attention to technical excellence and good design
- Simplicity
- Regular adaptation to changing circumstances

Using this method, after each iteration, the prototype is fixed and changes are made according to the feedback or data gathered during its release. A new and improved prototype is then created and released in the next iteration. The reason is that by having a demo often, any missing functionality, error or design flaw can be detected early and improved rather than having to wait until the testing and implementation time.

Agile unified process model's advantages are:

- Faster delivery
- Cost saving
- Simple

Figure 4: Flowchart of the project




Each prototype will be check to see whether its functionality is working or not. If it is working, then it is ready for the testing phase. After the testing phase is the process to compare the results of the load tests against the standard acceptable level of performance to see the game's performances

## 3.3 CONTENT DEVELOPMENT

Content development is the process of researching, writing, gathering, organizing, and editing information for publication on web sites that may contain graphics, audio, video and other medias for viewing and interacting in the Internet using a web browser. It is part of the content management that is a process or techniques to support the evolutionary of the web content development.

Web content development is often used to create applications to be used in the web-based environment. In this project, content development will be used to develop the online web-based strategy game on both platforms.

Content development process takes the following forms:

- Web planning
- Web analysis
- Web design
- Web implementation

### 3.3.1 Content Development Terminology:

- Web planning

  Web planning is the process of planning and determining the content of the web page, the target audience and deciding the purpose.
- Web analysis

  Web analysis is the process of analyzing the planning process and the decisions made.
- Web design

  Web design is the process of conceptualization, planning, modeling, and execution of electronic media delivery via Internet in the form of Markup language suitable for interpretation by Web browser and display as Graphical user interface (GUI)..
- Web implementation

  Web implementation is the process of implementing and testing the content of the web page in the Internet.

### 3.3.2 Benefits of content development:

Content development has many advantages and some of them are:

- Insight into how a Web site should be structured for aesthetics and navigation based on audience and purpose
- Information about processes and techniques used to build the content of a Web site
- An understanding of the multi-role, multi-disciplinary tasks involved in producing excellent Web site content
- A framework for Web content development which encompasses planning, analysis, design, implementation, and promotion.
- An appreciation for the unique information development challenges of the Web

## 3.4 LOAD TESTING

Load testing is the process of creating demand on a system or device and measuring its response.

To load test the Quantum Star, there are several steps to follow as a guideline on doing the load test as shown in Figure 1. The load test can help to identify the maximum operating capacity and to figure out the capabilities of Quantum Star based on the parameters that are tested.

### 3.4.1 Load Testing Steps

Step 1 - Identify Performance Acceptance Criteria

This step is to identify what are the parameters of Quantum Star that is chosen to be tested and their acceptable level of performance. The acceptable level of performance can either be defined before the test or by following to a standard that has already been established. This input will be used later in the result step.

The parameters and the acceptable level of performance selected for Quantum Star are:

- Throughput. (50 requests per seconds)
- Maximum user load / Scalability. (300 simultaneous users)
- Response time. (below 1 seconds)
- Page load time. (below 3 seconds)

Step 2 - Identify Key Scenarios

Identifying key scenarios is to identify the scenario that will affect the performance of Quantum Star. These scenarios will be used to create workload model and to design load tests. The key scenarios for Quantum Star are:

1. Multiple users simultaneously login into the game
2. Simultaneous users send requests at the same time to the server

Step 3 - Create a Workload Model

Creating a workload model is to determine how the load of the test is going to be distributed as users are affected by several scenarios such as:

- A user will typically pause between pages during a session. This is known as user delay or think time.
- A session will have an average duration when viewed across multiple users. It is important to account for this when defining the load levels that will translate into concurrent usage, overlapping users, or user sessions per unit of time.
- Not all scenarios can be performed by a new user, a returning user, or either; know who you expect your primary users to be and test accordingly.

Step 4 - Identify Target Load Levels

Identifying the load level's purpose in load testing Quantum Star is to ensure that the load tests can be used to predict or compare a variety of production load conditions depending on the parameters chosen.

Step 5 - Identify Metrics

Identifying metrics is to identify the metrics that will be monitored and observed depending on the parameters. Different parameters will have different metrics to be monitored to obtain the results.

The identified metrics in load testing the Quantum Star based on each parameter are as follows:

- Throughput: Throughput per minutes, Throughput per seconds and Throughput in kilobytes
- Scalability: Number of users and Error rates
- Response Time: Average response time and Maximum response time
- Page Loading Time: Total loading time

Step 6 - Design Specific Tests

Using the scenarios, key metrics, and workload analysis, specific tests to be conducted can be designed. Each test will generally have a different purpose, collect different data, include different scenarios, and have different target load levels. The key is to design tests that will help collect the information it needs in order to understand, evaluate, or tune the application.

Each test is designed by taking into account the software that will be used to run the test. The test's design for each parameter is:

- Throughput: 100 simultaneous users sending requests
- Scalability: Increasing number of users until errors are displayed
- Response time: 100 simultaneous users sending requests
- Page load time: Loading the page inside a browser

Step 7 - Run Tests

The load tests are conducted using several softwares where each of the software has its functions in collecting the tests results and displaying them. The tests are conducted by inputting the identified loads and running the softwares to see the results. The softwares used to test each parameter are as follows:

- Throughput: JMeter
- Scalability: JMeter
- Response time: OpenWebLoad
- Page load time: Load Time Analyzer

Step 8 - Analyze the Results

After the results are collected, each parameter tests' results are then compared to the defined performance acceptance criteria as stated in Step 1. From this comparison, it can be determined how each parameter is performing against the defined performance acceptance criteria.

This will show whether the parameters of Quantum Star chosen is justified in term of performance. The comparison is done as follows:

- Throughput: against defined value
- Scalability: against defined value
- Response time: against R. D. Miller standard
- Page load time: against Microsoft standard

# CHAPTER 4
# RESULT AND DISCUSSION

The project is done in two phases, the first one is developing the online text-based game and the second phase is to run a load test on the game and compare the data obtained from the tests against the standard acceptable level of performance. The game will be an online text-based strategy game that can be accessed using the Local Area Network (LAN) at first and then in the Internet.

The game is hosted on a webserver so that users can connect to it through the network and the internet. The webserver handles users' requests to the server when playing the game. Apache webserver is used in the project as it is a free, common and easy to use and configure webserver.

## 4.1 GAME DESIGN

The game developed is an online text-based strategy game. The interface and layout of the game are developed using HTML and PHP. The database of the game are handled using MySQL and phpMyAdmin.

The game is called Quantum Star: Generations, an online text-based space strategy games that utilizes turns-based strategy to run. The game is developed based on a template obtained from the internet that has undergone lots of fixes and changes to the functionality to make them work properly in order to be tested.

Some of the fixes and changes are done in the creating a user function where previously the function is not working as the data sent from the signup form conflicts with the SQL table structure and therefore the data query to the SQL has to be rewritten.

Another example involves fixing most of the links or queries from the PHP to the SQL as most of them are broken because of several factors that include missing arguments, wrong data are retrieved from the forms and sent to the SQL to update and the PHP forms are not capturing the input from the players.

Other than that, an improvement on the appearance of the game layout was done mostly by optimizing and rewriting the HTML codes of the web pages such as the table structure, links, information displayed and the CSS (Cascading Style Sheet) used in the development of the game. But, all these changes are done carefully so as not to make the game's interface differ from the initial interface.

The objective of the game is for players to manage a fleet of space ships that wanders within the game's universe and collecting resources while building up their army and going against other players.

To start playing, a player must first create an account that will require their basic profile and once done, the player can login into the game using the created account.

Figure 9: Signup Page

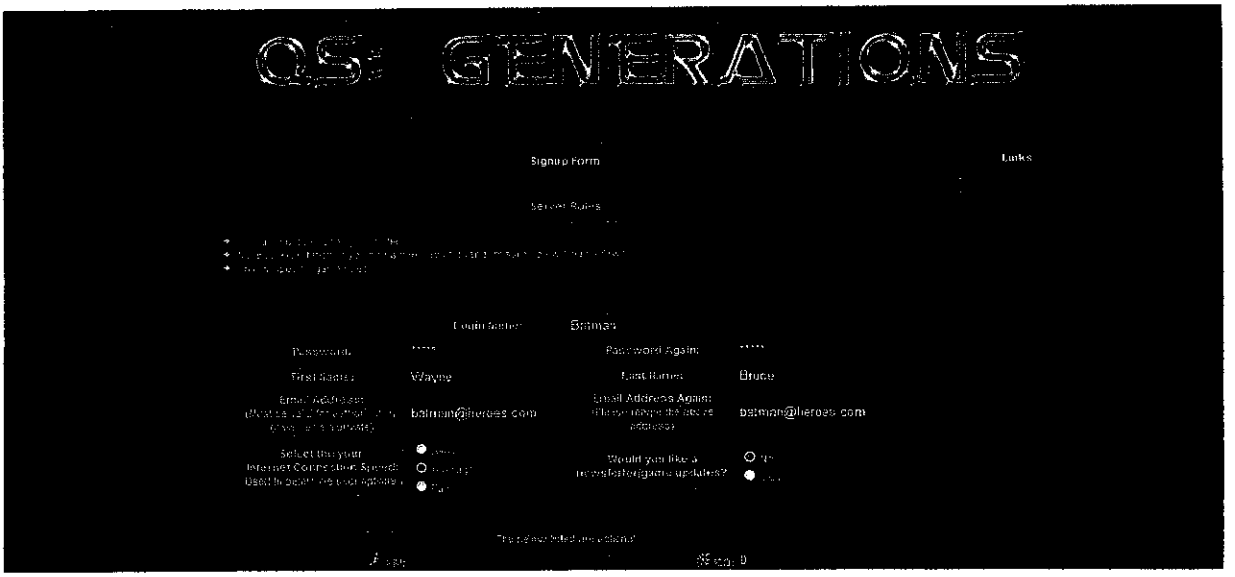
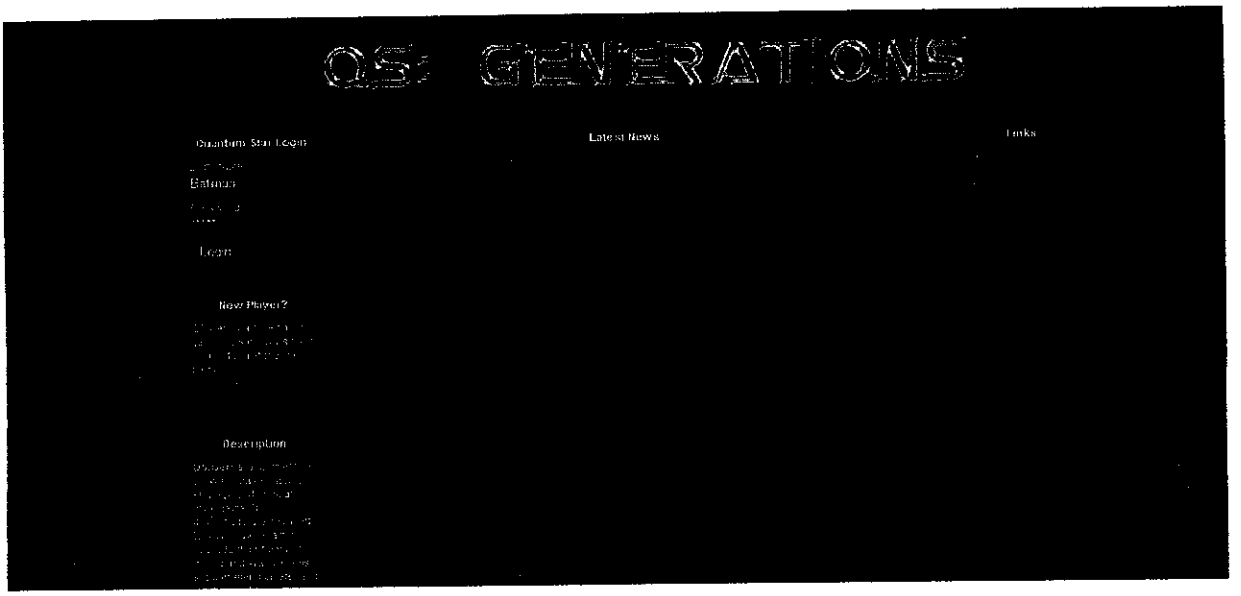
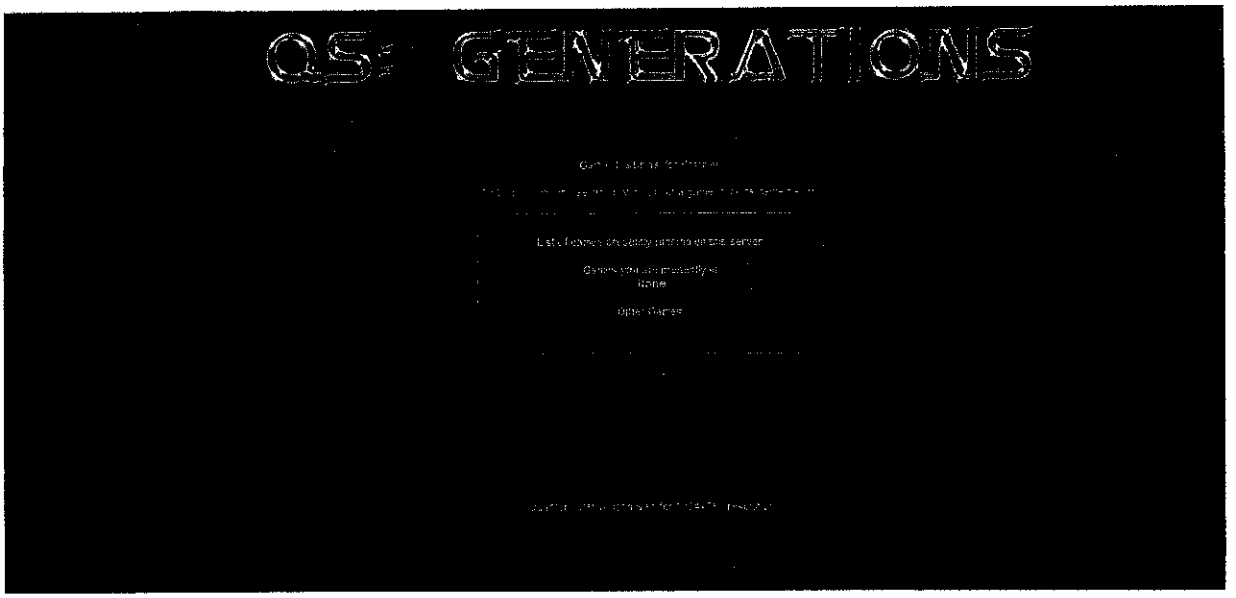


Figure 10: Login Page



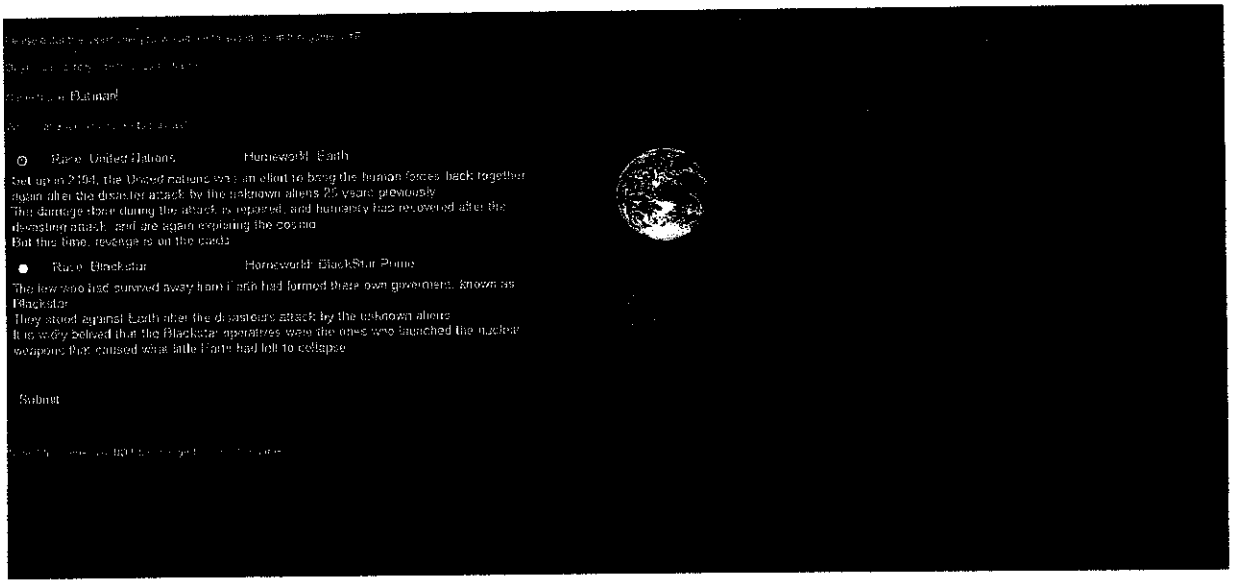

After a player has login into the game, a page where all the available game listings for the player to choose will be loaded up and the player must select a game listed there to start playing.

Figure 11: Game Listing Page



Then, player will need to select their race or faction that they want to use in the game which is either the United Nations or the Blackstar faction and also players can specify the name that they will be shown as in the game that they are participating.

Figure 12: Race Selection Page




After choosing a race or faction, player will come to the home page of the game. This page contains almost the entire summary of the player's condition including the number of available turns, amount of money available, list of the player's fleet status, list of enemy's fleet in the current Star System, resources available for mining in the current Star System and games related news.

From this page, players can take a quick look on the situation and condition of the current situation and then select the best course of action by going to other pages to continue their action to either start mining the resources, attack the enemy, warps to another Star System and other options.

Figure 13: Home Page

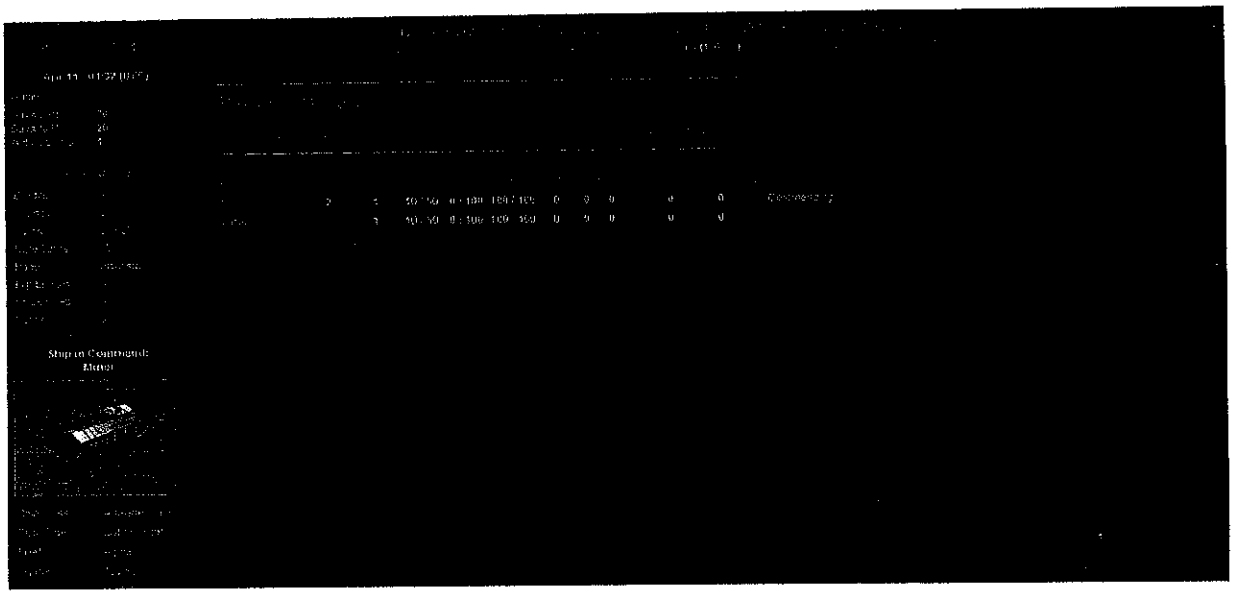
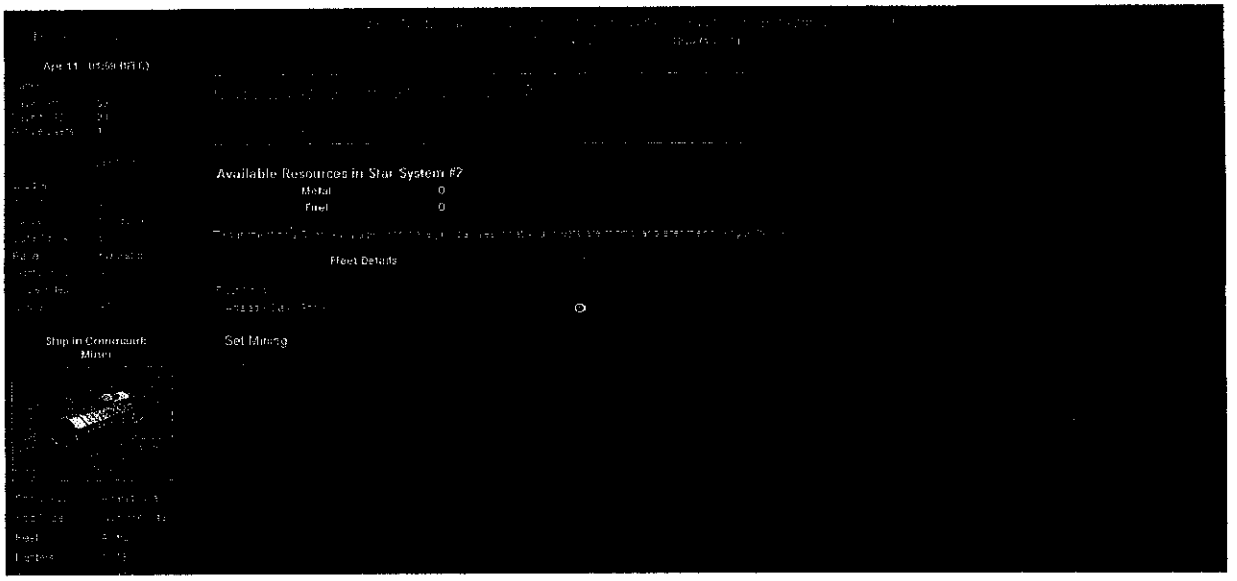
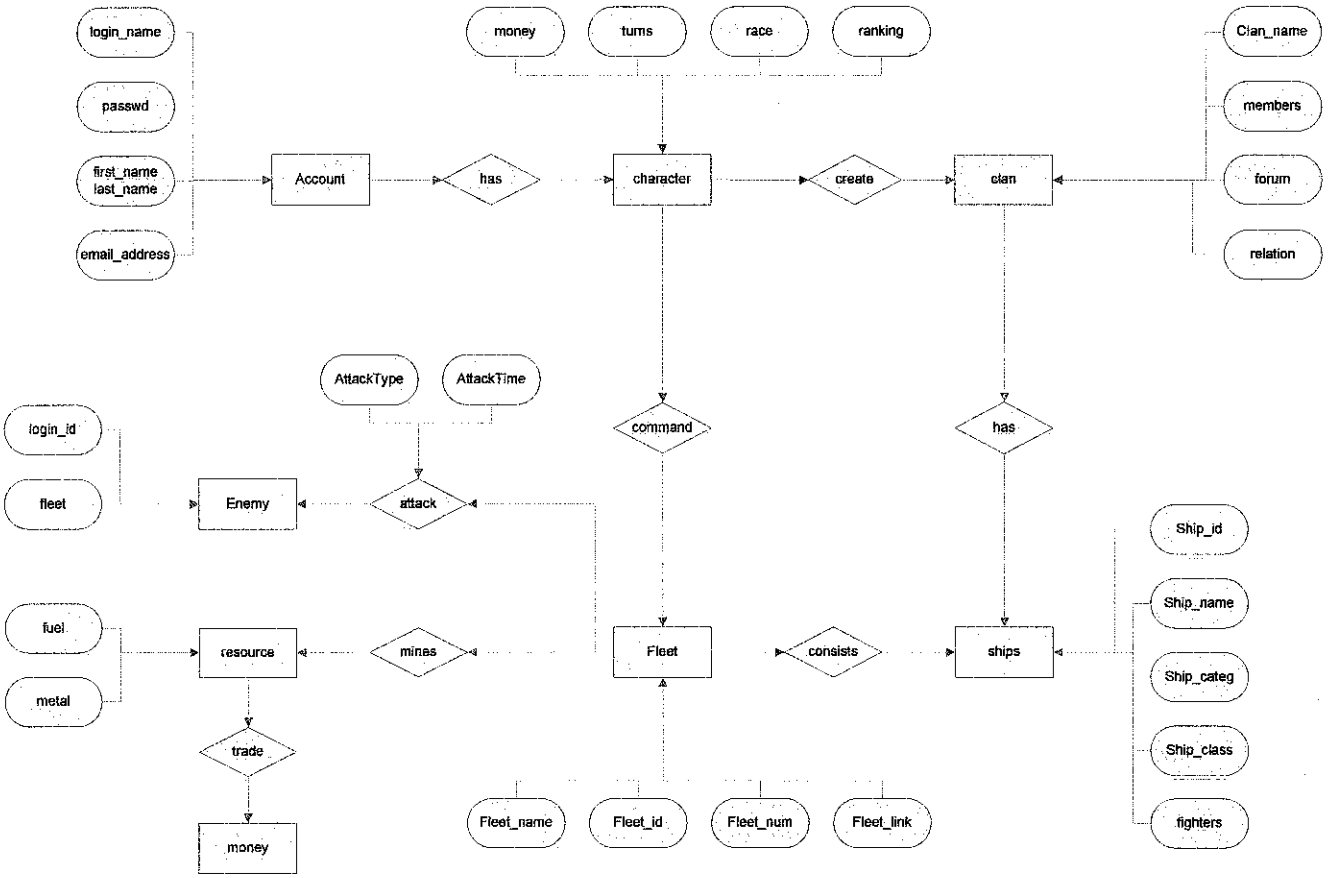


Figure 14: Mining Page



An ERD (Entity Relations Diagram) of the game's architecture shows how the elements and variables of the game are related and functions as there are quite a number of elements and variables that are involved in developing this game.

*Figure 15: ERD of the online text-based strategy game*



## 4.2 TESTING PHASE

After Quantum Star has been developed, the load tests are run on the game to obtain the data on the performances of the game. The load tests are conducted using several softwares and methods. The tools used are JMeter, OpenWebLoad and Load Time Analyzer.

The parameters that are chosen to be observed are:

- Throughput
- Scalability
- Response Time
- Page Loading Time

Throughput is a measurement of how much requests per minute that the web application can handle. This parameter shows whether the web application can handle heavy load in which multiple users will make requests to the web application and the ability of the web application to handle the requests are recorded and plotted into a graph using JMeter.

Scalability is a desirable property of a system, a network, or a process, which indicates its ability to either handle growing amounts of work in a graceful manner, or to be readily enlarged. In this case, scalability refers to the ability of the web application to handle a maximum number of users before the web application starts to generate errors to the clients. Multiple sets of load tests with different load setting are used to record the maximum number of users that the web application can handle.

Response time is the time a system or functional unit takes to react to a given input. The response time of a task or thread is defined as the time elapsed between the dispatch (time when task is ready to execute) to the time when it finishes its job (one dispatch). By using OpenWebLoad to start sending requests to the server and then monitor the time the server takes to reply to the client, the response time of the web application can be obtained.

Page load time is the measurement of how long does the particular web page takes to be fully loaded into the browser. This is important as text-based games should give a quick response time due to the minimum use of graphic in the interface. This parameter is tested using the Load Time Analyzer which can record the time taken to fully load a web page and plots the result into a graph.

All these parameters will then be compared against the standard acceptable level of performance as specified by certain people and organization.

For throughput and scalability, no standard acceptable level of performance are found as these two parameters are usually a predetermined parameter values that is set before the start of the project to be the benchmark to test the performance of the web application. In this comparison, the acceptable level of performance for throughput is determined to be fifty requests per second as this is quite a popular benchmark number used in other projects. The scalability parameter is determined to be at three hundred simultaneous users as this number is quite a large number of users.

For response time, based on R. B. Miller (1968), the standard acceptable level of performance for this parameter is below one second. This standard has been establish for forty years and is widely accepted as the standard for response time parameter and is said that it will not change even with the advancement of the current technology.

For page loading time parameter, a study done by Microsoft until 2008 has establish a standard for page loading time and the standard for loading common pages are set to be under three seconds.

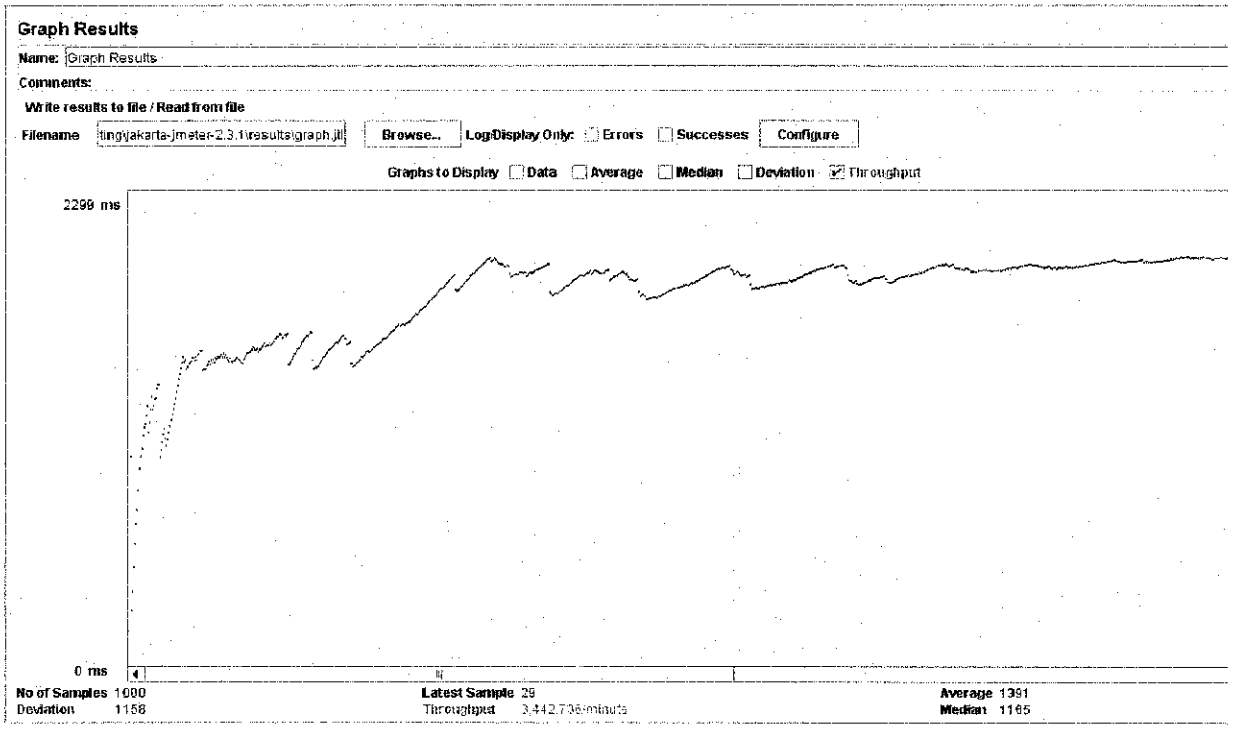
## 4.3 TEST RESULTS

### 4.3.1 Throughput

To measure the throughput parameter, the software used is JMeter. JMeter is configured to emulate 100 virtual users that will make requests to the web application simultaneously without looping with 0 seconds interval between users meaning that each initial user requests will be started simultaneously.

The result is as follows:

Figure 16: JMeter Graph Result




The graph in Figure 16 shows the throughput of Quantum Star against the time it takes to produce the throughput in milliseconds.

Table 1: Throughput analysis

| Throughput (100 users) | | | |
|---|---|---|---|
| **Tries** | **per minute** | **per seconds** | **kilobytes** |
| 1 | 3326.311 | 55.4 | 53.98 |
| 2 | 3457.018 | 57.6 | 56.1 |
| 3 | 3350.832 | 55.8 | 54.37 |
| | | | |
| **Average** | 3378.054 | 56.3 | 54.8 |

Table 1 shows the throughput analysis of Quantum Star after three tries and the average throughput in three categories; throughput per minutes, throughput per seconds and the throughput in kilobytes.

Table 2: Comparison of throughput analysis against acceptable performance

| Throughput (requests/sec) | | |
|---|---|---|
| **Quantum Star** | **Acceptable** | **Difference** |
| 56.3 | 50 | 6.3 |

Table 2 shows the comparison between the throughput analysis of Quantum Star against the acceptable performance defined before the tests are conducted.

Figure 17: Comparison between throughput and acceptable performance

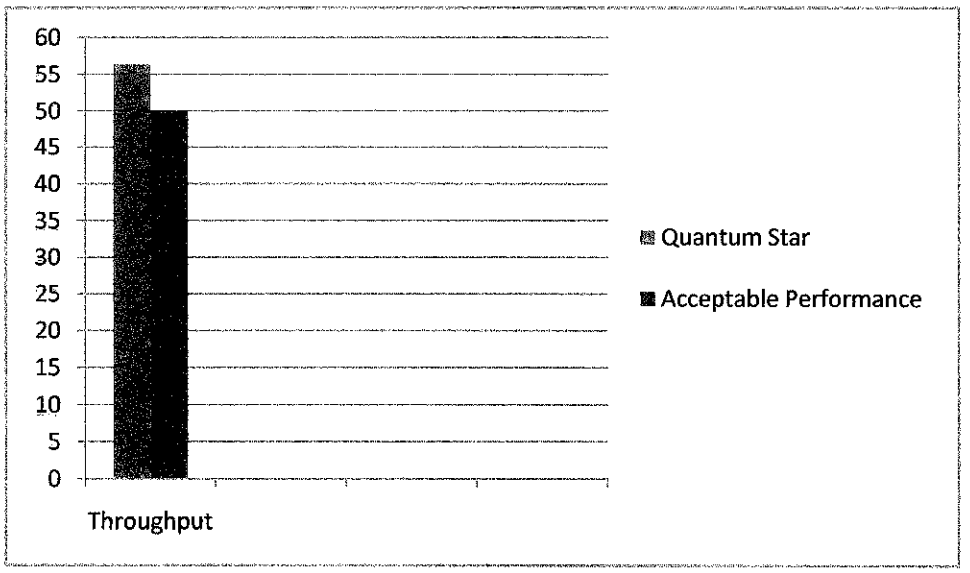


Figure 17 is a graphical representation of Table 2 that show the comparison between the throughput analysis of Quantum Star and the acceptable performance. From the comparison, it can be seen that the throughput of Quantum Star is higher than the defined acceptable level of performance by 6.3 requests per seconds. This is good as higher throughput means that Quantum Star can handle more requests from user than accepted.

### 4.3.2 Scalability

JMeter is used to measure the scalability parameter of the web application. The tests are done by starting at a low number of users that are gradually increasing as long as the tests does not show that the web application is generating errors that is sent to the client.

The result is as follows:

Table 3: Scalability analysis

| Scalability | |
|---|---|
| **Users** | **Error rate %** |
| 300 | 0 |
| 400 | 0 |
| 500 | 0 |
| 550 | 1.82 |
| 600 | 6.83 |
| 700 | 10.57 |

From Table 3, it is shown that Quantum Star only starts to generate errors when the number of users reached 550 users. When comparing this to the acceptable level of performance defined earlier which is 300, the following observation are acquired:

Table 4: Comparison of scalability analysis against acceptable performance

| Scalability (number of users) | | |
|---|---|---|
| **Quantum Star** | **Acceptable** | **Difference** |
| 550.0 | 300 | 250 |

Table 4 compares the scalability analysis of Quantum Star against the defined acceptable level of performance.

Figure 18: Comparison between scalability analysis and acceptable performance

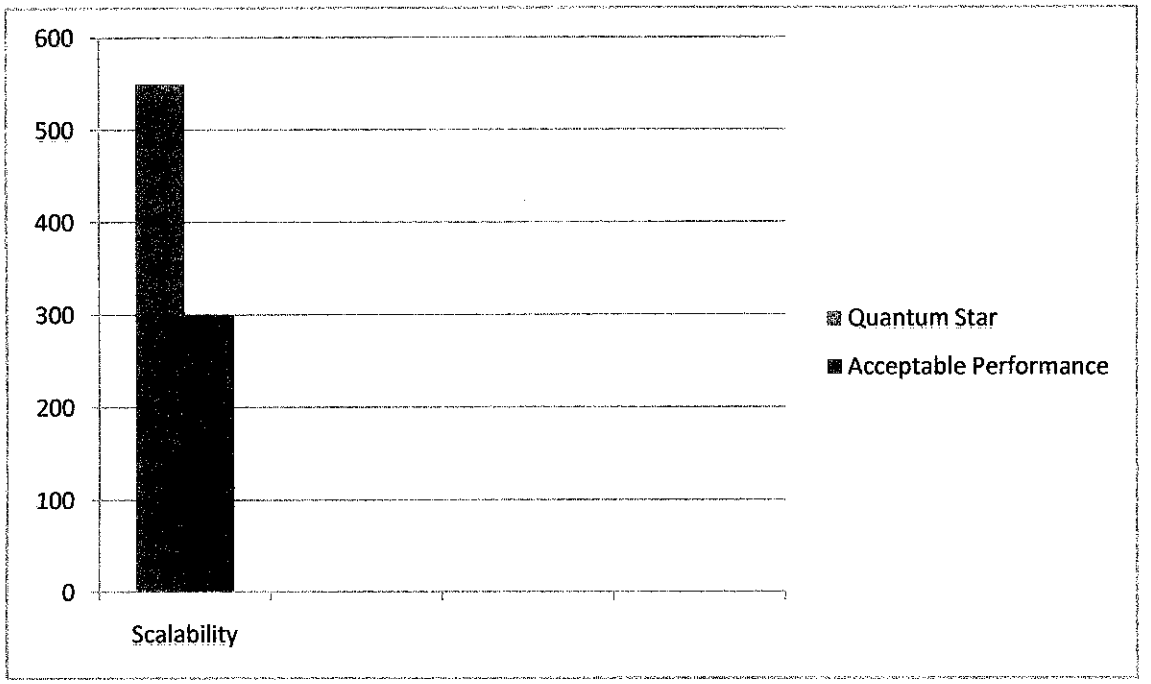



Figure 18 is a graphical representation of Table 4 that compares the scalability analysis of Quantum Star against the acceptable performance defined earlier.

The difference of 250 users shows that Quantum Star managed to accommodate the target number of user which is 300 and is able to support an additional 250 users before it starts to produce errors.

### 4.3.3 Response Time

Response time is measured using OpenWebLoad. This software will emulate 100 virtual users that will send requests to the web application and the software will monitor and capture the time taken for the web application to reply as the response time.

The result is as follows:

Figure 19: OpenWebLoad Result

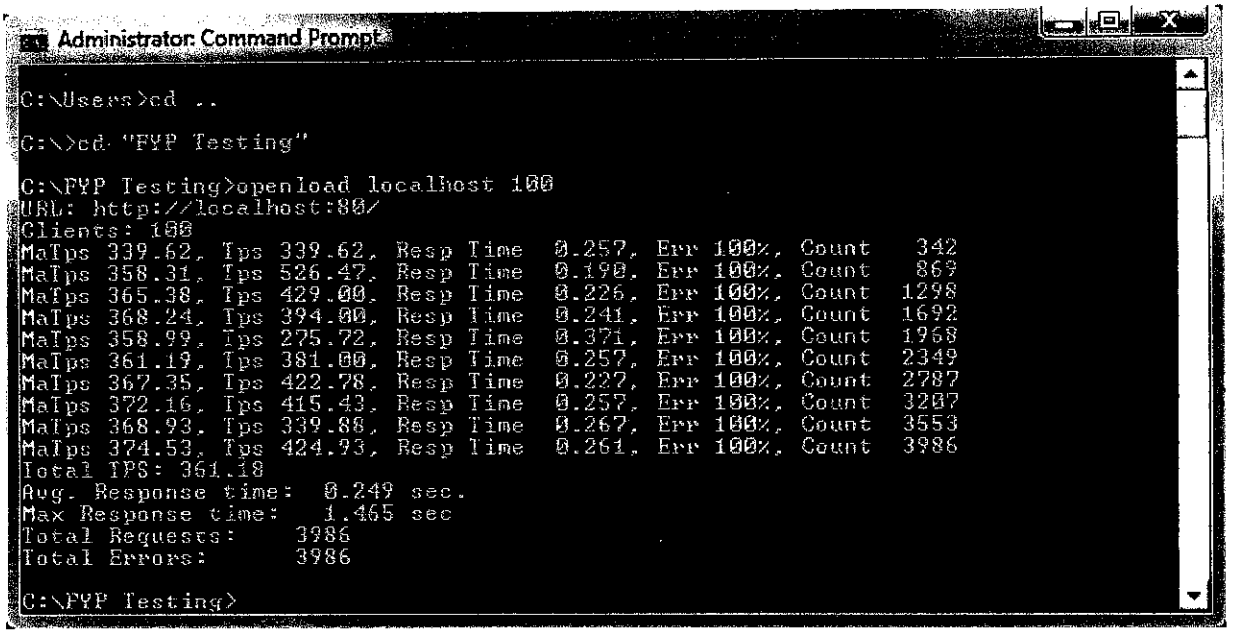



Figure 19 shows a screenshot of the OpenWebLoad program displaying the result of the test inside a command prompt. The result shows that the average response time is 0.249 seconds and the maximum response time is 1.465 seconds.

Table 5: Response time analysis

| Response Time (100 user) | | |
|---|---|---|
| **Tries** | **Average Time** | **Max Time** |
| | | |
| 1 | 0.249 | 1.465 |
| 2 | 0.234 | 1.227 |
| 3 | 0.225 | 1.408 |
| | | |
| **Average** | 0.236 | 1.367 |

Table 5 shows the result of the response time analysis of Quantum Star that is done in three tries. The average response time of the tests is 0.236 seconds and the maximum response time is 1.367 seconds.

This result is the compared to the standard set by R. D. Miller in 1968 which is below 1 second.

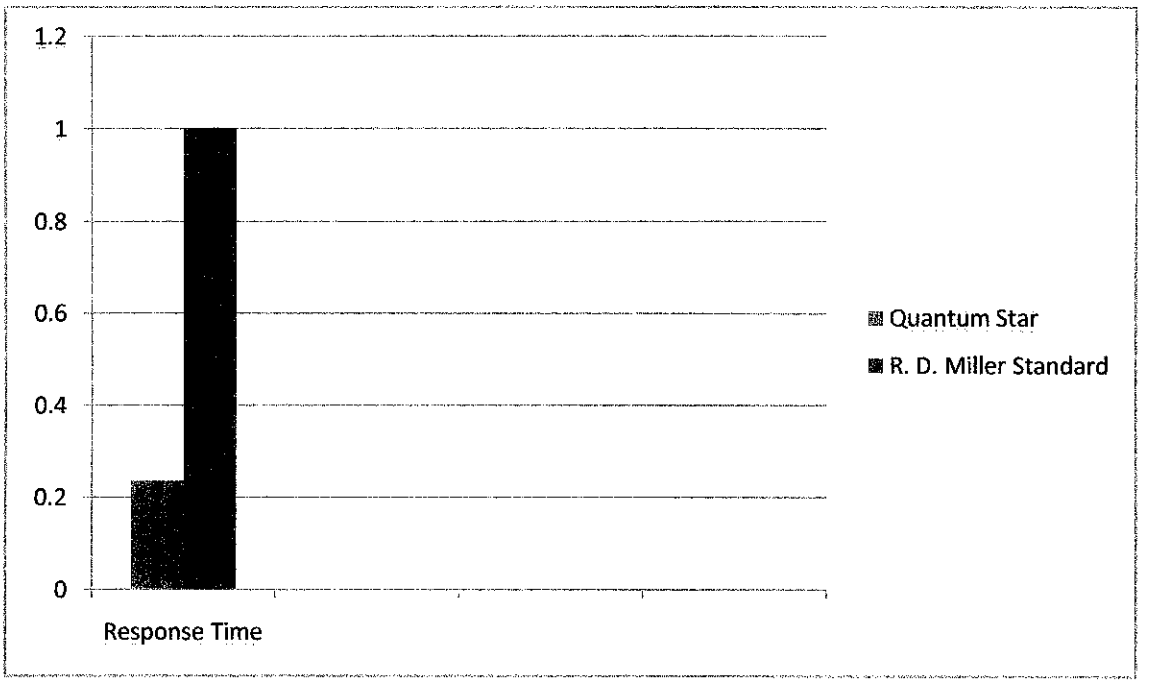
Table 6: Comparison of response time analysis against standard acceptable performance

| Response Time (seconds) | | |
|---|---|---|
| **Quantum Star** | **R. D. Miller Standard** | **Difference** |
| 0.236 | 1 | 0.764 |

Table 6 shows the comparison between the response time analysis of Quantum Star and the standard acceptable level of performance as defined by R. D. Miller in 1968.

The graphical representation of Table 6 is shown in Figure 20 that shows the difference between the Quantum Star response time and the standard acceptable level of performance of response time as defined by R. D. Miller.

Figure 20: Comparison between response time and R. D. Miller Standard




From Figure 20, it is shown that the difference of response time is 0.764 seconds lower than the standard acceptable level of performance as defined by R. D. Miller. That means that this parameter response time is very good.

### 4.3.4 Page Load Time

This parameter is measured using the Load Time Analyzer. The Load Time Analyzer is embedded into the browser and it will capture the total time taken to fully load the web application into the browser and produce a graph with details on the loading time.

The result is as follows:

Figure 21: Load Time Analyzer Graph Result

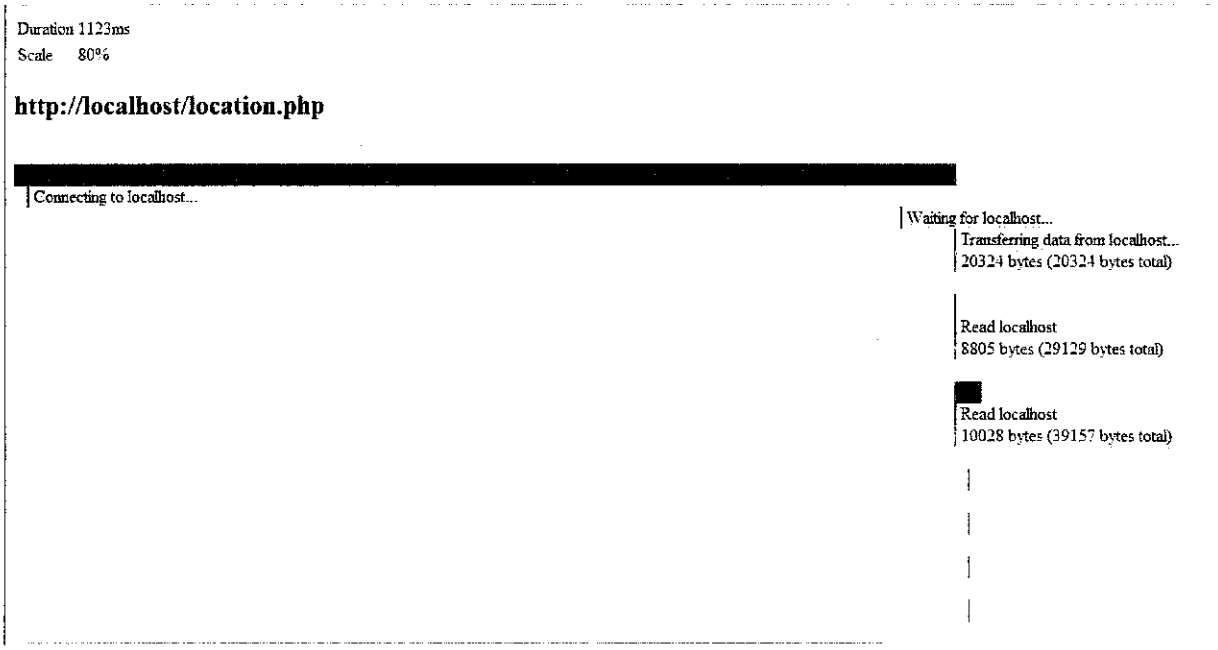



Figure 21 is a screenshot of the graph produced by the Load Time Analyzer software when tested on the Quantum Star. It shows the time it takes to fully loaded a page and the graph is the showing the load time of every element of the tested page with detailed description of the graph shown on the right as shown in Figure 22.

Figure 22: Detailed Load Time Analyzer Graph Result

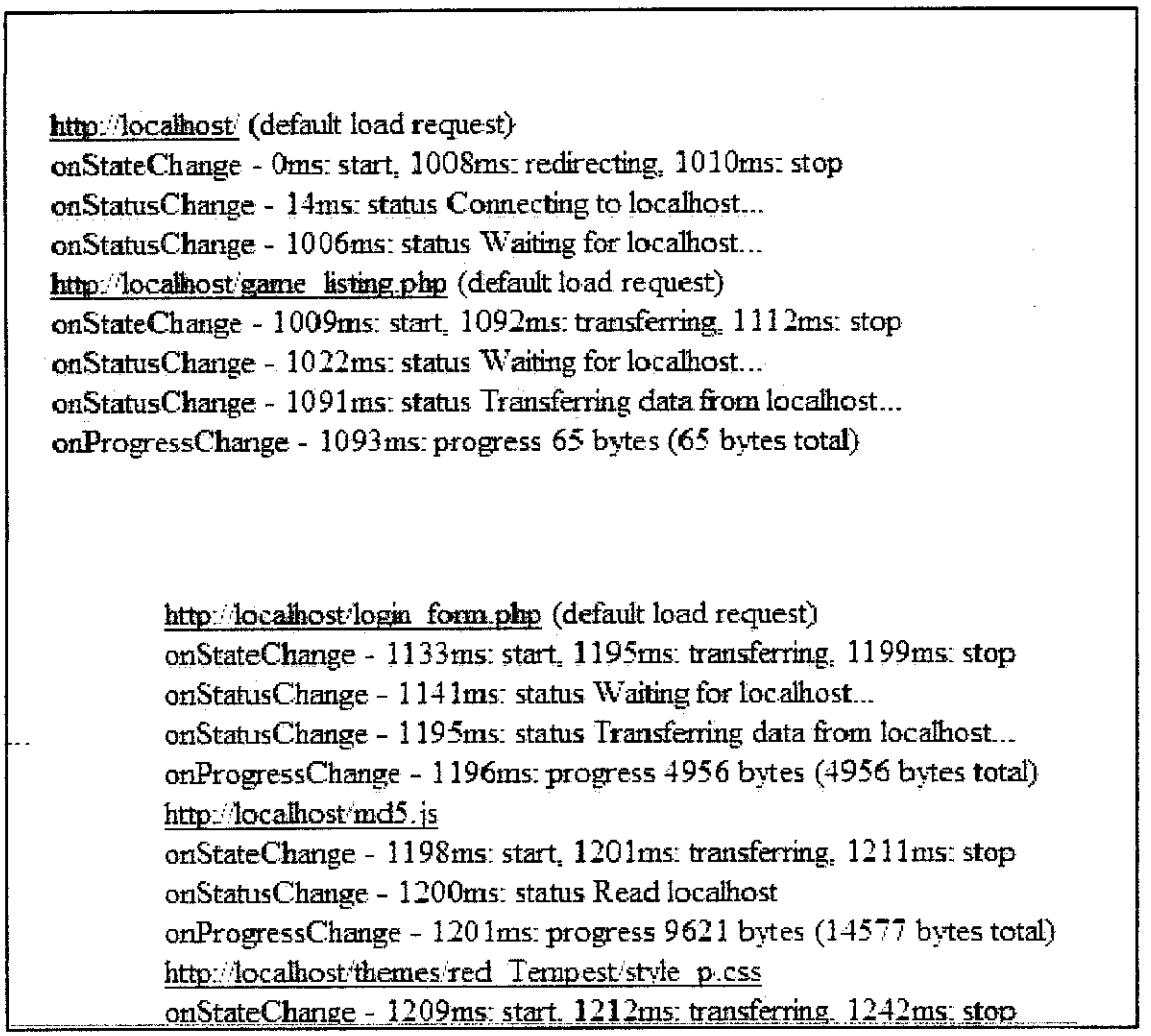



Figure 22 shows detailed descriptions on the graph produced by the Load Time Analyzer by stating the amount of time that a page takes to fully load every element inside it.

Table 7: Page loading time analysis

| Page Loading Time | |
|---|---|
| **Tries** | **Time (ms)** |
| 1 | 1123 |
| 2 | 1170 |
| 3 | 1170 |
| | |
| **Average** | 1154.3 |

Table 7 shows the page loading time analysis of Quantum Star obtained from the Load Time Analyzer after three tries.

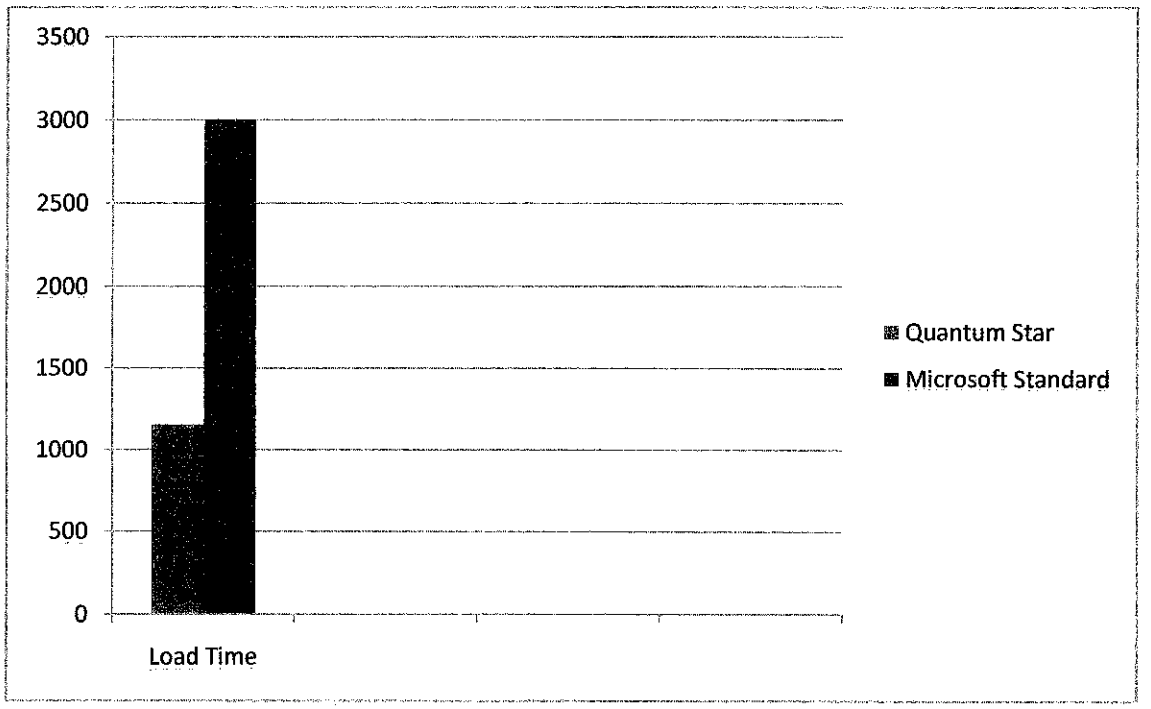
These results are then compared to the standard acceptable level of performance as specified by Microsoft to produce this result:

Table 8: Comparison of page loading time analysis against standard performance

| Page Load Time (milliseconds) | | |
|---|---|---|
| **Quantum Star** | **Microsoft Standard** | **Difference** |
| 1154.3 | 3000 | 1845.7 |

Table 8 compares the average page loading time of Quantum Star against the standard acceptable level of performance as defined by Microsoft. The graphical representation on Table 8 is shown in Figure 23.

Figure 23: Comparison between page loading time and Microsoft standard



The result from Figure 23 shows that the difference in the response time is 1845.7 milliseconds which is lower than the Microsoft standard. This shows that Quantum Star manages to fully load its pages in a time frame that meets the Microsoft standard.

## 4.4 DISCUSSIONS

The project was conducted in two semesters in two parts. The first part was mainly on studies, preliminary research and software and game design. The second part focuses more on software programming and testing.

At the end of the project, all the objectives are able to be achieved although there are still some problems with the project. For example, the development of the game did not manage to fully complete the game with all the planned features and functions. Although most of the basic and important features and functions needed to run the game is completed, the game cannot be tested for its full capabilities at the moment. To solve this problem, the solution is either having more time to develop these unfinished functionalities or just remove them from the game completely.

Another problem is due to the difficulties faced in completing the game, a thorough test and check has not been done to check the functionality of the game thoroughly in order to avoid unsuspecting bugs to emerge. More time are needed if a thorough test of the game's functionality is to be done.

The testing also face a problem where the users used to test the game's performance parameters are mostly virtually created user due to difficulties in assembling a large number of real life users to test the web application simultaneously. This situation might cause a variation in the results as the test are done virtually and not in real time. To overcome this problem, a session could be arranged where a lot of users are called to participate in testing the game's performance.

However, even with all these problems, the game seems to be functioning well so far and the results of the load tests are also convincing enough for this project to be considered a successful project although there are some areas that can be improved as stated above.

# CHAPTER 5
# CONCLUSION AND RECOMMENDATIONS

## 5.1 CONCLUSION

This project shows that the developed online text-based game is able to meet the requirement of the standard acceptable level of performance. This shows that the developed online text-based game is can provide an acceptable level of performance to all players that plays the game regardless of their network connection's strength.

This project enables users of the internet to know what does the acceptable level of performance that an online text-based game should be able to offer for the users to be able to experience a nice and smooth user experience and having fun at the same time.

There are certain parts that still need to be developed mainly the CGI for the web pages in order to make the web pages to be able to function normally.

As a conclusion, this project is a success as it managed to meet all the objectives and solve the problem although there are still some improvements that can be made especially to the game design.

## 5.2 RECOMMENDATION

This project can still be improved in certain areas. First of all, although most of the basic functions of the game needed to run the game are functioning, some of the additional features and functions are yet to be completed.

Next, the testing phase could be improved by using other tools to carefully control the testing environment in order to achieve an even more precise set of data to be used in the comparison.

Also, a comparison of the developed online text-based game against other online text-based games will strengthen the result of the comparison as the game is compared to something that is quite similar to its nature.

# REFERENCE


1. Online calibrated forecasts: Memory efficiency versus universality for learning in games
   Mannor, S., Shamma, J.S. (et al.)
   Machine Learning, Springer US, 27.09.2006, vol. 67, no. 1, pp. 77-115

2. A dynamic priority approach to reducing delay in interactive TCP connections
   Dimopoulos, P., Zeephongsekul, P. (et al.)
   Telecommunication Systems, Springer US, 20.12.2006, vol. 34, no. 1, pp. 59-70

3. Browser-based applications: popular but flawed?
   Silver, M.S.
   Information Systems and e-Business Management, Springer, 16.02.2006, vol. 4, no. 4, pp. 361-393

4. Miller, R. B. (1968). Response time in man-computer conversational transactions. Proc. AFIPS Fall Joint Computer Conference Vol. 33, 267-277.

5. Plan for Software Boundaries, Microsoft;
   http://technet2.microsoft.com/WindowsServer/WSS/en/library/

6. Usability Engineering
   Jakob Nielsen
   Morgan Kaufmann, San Francisco, 1994. ISBN 0-12-518406-9

7. Load Testing Web Applications; http://msdn2.microsoft.com/en-us/library/bb924372.aspx; September 2007

8. JMeter; http://jakarta.apache.org/jmeter/index.html;

9. OpenWebLoad; http://openwebload.sourceforge.net/;

10. Load Time Analyzer; https://addons.mozilla.org/en-US/firefox/addon/3371;

11. Quantum Star; www.quantum-star.com/;

12. http://en.wikipedia.org/wiki/Iterative_and_incremental_development#Life-Cycle

# APPENDIXES

APPENDIX A: ERD of the Online Text-Based Strategy Game

APPENDIX B: JMeter Sample Page

APPENDIX C: Load Time Analyzer Sample Page