NETWORK MONOPOLY

Tan Kean Yeap
WEK 990064
2001/2002

Supervisor
Dr. Mazliza Othman

Moderators
Cik Nurul Fazmidar Mohd Noor
Puan Hannyzzura Pal @ Affal

Faculty of Computer Science & Information Technology
University of Malaya

Acknowledgements

I would like to express my heartiest thanks to Dr Mazliza Othman for giving me the opportunity to work in her project and for the endless support, timeless supervision and guidance rendered throughout the duration of the project.

A big thanks also goes to my project moderators, Cik Nurul Fazmidar Mohd Noor and Puan Hannyzzura Pal for their suggestions and comments.

My heartfelt gratitude also go out to the lecturers of the faculty of Computer Science and Information Technology for their dedication in giving valued lectures on computing over the years.

Nor forgetting my fellow friends for their ideas and advice throughout the preparation of the project.

Last but not least, a very special word of gratefulness to my parents for their endless love and encouragement.

Abstract

The number of computers in use worldwide, according to the International Data Corporation, is in excess of 100 million. Moreover, because of the expanding memory and processing power of these computers, users can put the machines to work on new kinds of applications and functions. Accordingly, the pressure from the users of these systems for ways to communicate among all these machines is irresistible. The demand for connectivity is manifested in many areas. One such area is about the creation of dynamic network entertainment.

Multiple-party gaming is a fascinating and multifaceted topic to be researched on. This thesis steps towards developing a networked board game, supporting multiple players. The main aspect of this thesis is to produce a network monopoly system, which process multiple game sessions concurrently, enabling users to play the graphical game with people within the network. The algorithm design and implementation method must not only be workable, but also highly efficient in terms of execution speed and respond time. With the powerful networking features of Java, it is expected that this project would produce a "real" game engine.

The possible applications of multimedia-networked system are endless, as the demand to support interactive communication among groups of users has increased significantly. The network solution applied in this game system permits only low level of communication, particularly for sending and receiving limited amount of messages within network. The category of networking could serves to be a small foundation from which to base the future scenario of other reliable multicast and real-time web based gaming system.

Table of Contents

CHAPTER 1
INTRODUCTION

Chapter 1: Introduction

1.1 Project Definition

Network monopoly is a game that is implemented using the client/server approach. The program consists of a server application that allows client system to connect to the server and play the game. The server application is the core of the game and must be running for the clients to work. The client is an application that resides in others computer that connected to the server computer. Because at least two players are required to start a game, the client is in a wait state until another player comes along.

Each player plays the game with other players by sending messages through network. The basic functionality of a game client is to connect to the server and communicate the user's actions and receives the game state information from the server, and updating itself accordingly. The game also involves displaying the graphics and managing the entire game interface for the users.

One of the coolest things about a network game is that your opponent can be anywhere, anywhere at all! You can play with the person next to you or your opponent could be in the next room, as long as they're connected to the network.

Monopoly is the best-selling board game in the world, licensed or sold in 80 countries and produced in 26 languages including Croatian. (see Appendix A)

It was created in 1934, during the Great Depression. The game immediately became a success and when the original creator could not keep up with demand he sold rights to the game to Parker Brothers. It has become extremely popular and it is played by people all over the world.

Network monopoly also features a "chat" window, which allows players to communicate with each other during the game play. The users will be able to "say" things to others players. This leads the way to generate a strong level of interaction among users.

1.2 Project Motivation

This section focuses on the motivation of developing this network monopoly.

1.2.1 Why network game?

Computer games seldom provide a human opponent, and so they lack the social or interpersonal element. On the other hands, network games provide more interactive element than stand alone computer games offer. Interaction is important for several reasons. First, it injects a social or interpersonal element into the event. It transforms the challenge of the game from a technical one to an interpersonal one. Solving a cube puzzle is a strictly technical operation; playing chess is an interpersonal operation. In the former, one plays against the logic of the situation; in the latter, one uses the logic of the situation to play against the opponent.

Second, interaction transforms the nature of the challenge from a passive challenge to an active challenge. A puzzle will always present the player with exactly the same challenge. But a game opponent reacts to player's actions, and presents different challenges in each game.

1.2.2 Why Monopoly?

Since monopoly is the best-selling board game in the world, it provides a great framework for human interaction with all the wacky wheeling and dealing. Some surveys have proven that monopoly can be use as a simulation game in the classroom to teach financial accounting.

Since it is the most popular game in the world, most players are likely to be familiar with the game and this will reduces the time taken by players to learn the game before joining the game.

1.3 Project Purpose

The purposes of the project are:

- Develop a game that is familiar to most us.
- Develop a game that is suitable for a large range of people.
- Develop a game that can play by at least two players.
- Develop a game that can provide education besides game playing.
- Develop a game that can provide social interaction among players.
- Develop a game that consists of multimedia elements.
- Develop a game that can provide a great activity for people to spend their time.

1.4 Project Objective

The fundamental objective is to provide the players with enjoyment. This is not just any means of enjoyment but it also meets the objectives stated below:

- Games can be used as social lubricants (especially by adults). The game itself is of minor importance to the players; its real significance is its function as a focus around which an evening of socializing will be built. As the players contort to fulfill the game requirements, they inevitably make physical contact with each other in an innocent and foolishly humorous ways. Social interaction is, thereby, fostered.
- It provides a means of overcoming social restrictions, at least in fantasy. The player's role is itself socially acceptable, but the actions taken are discouraged in real life. Monopoly encourages players to engage in what the US Federal Trade Commission delicately calls "predatory trade practices".
- It provides a great framework for human interaction with all the wheeling and dealing.
- It provides a way of education. Surveys have proven that monopoly can be used as a simulation game in the classroom to teach financial accounting (see Appendix B). It can offer the following advantages:
  - Increased ability to recall factual knowledge and improve problem-solving skills.
  - Require flexibility in thinking as students adapt to a dynamic environment.
  - Give students, through the "wheeling and dealing" simulations, intensive practice in verbal and written communication.
  - Benefit students with varying skills and experience because participants can play at their own level.

- Offer the potential for students to attribute greater value to accounting information in the decision-making process.
- Offer students participating in simulation games experience greater attitudinal change than those engaging in more traditional learning environments.

- ❑ Provide a great way to play with friends and family.

1.5 Project Scope

The scope of the project outlines the limitations of the application and the boundaries set upon the application.

- ❑ The core of a network game revolves around a client/server communication strategy. In fact, the design of the game can be divided cleanly into the client side and the server side. These two components are logically separated, communicating entirely through a game protocol defined between them.
- ❑ The server side of the game acts almost like a referee, managing the different players and helping them communicate effectively. More specifically, a game server takes on the role of handling the network connection for each player, along with managing the flow of the game between the players and informing each player of the state of the game, while modifying the state according to each player's turn.
- ❑ The basic responsibility of a game client is to connect to the server and communicate the user's actions, along with receiving game state information from the server and updating itself accordingly. Of course, along with this comes the

responsibility of displaying the game graphics and managing the entire game interface for the user.

- For a network game to work, you must have the game server always running in the background. Because two players are required to start a game, the client is in a wait state until another player comes along. The maximum number of players in a single game is 4.
- The game itself consists of two co-existing major components, the game area and the chat area. The game area displays the monopoly board while the chat area instead is a common test field, which allows players to chat while playing the game.

1.6 Project Outcome

Network Monopoly focuses on developing a gaming system that can be play within network. At the end of the project, the gaming system is expected to have the following characteristics:

- Support multi-players gaming.
- Support network game play.
- Able to process multiple game requests at once.
- Allow players to communicate with each other during the game play by sending short messages.
- Consists of multimedia elements, such as audio, graphic, animation, etc.
- Able to run in multiple platforms.
- Consistent, stable and user friendly.

- Reliable, robust, accurate.

1.7 Project Schedule

The project schedule is the operating timetable of the project. It serves as the fundamental basic for monitoring and controlling project activity. In a project environment, proper scheduling functions is of paramount importance because projects lack of continuity of day-to-day operations and often present much more complex problems of coordinator. The project schedule for the network monopoly is shown in the table below:

	Task Name	July 2001	Aug 2001	Sep 2001	Oct 2001	Nov 2001	Dec 2001	Jan 2002	Feb 2002
1	Literature Survey	▨							
2	System Analysis	▨	▨						
3	System Design	▨	▨						
4	Construction of Prototype				▨				
5	Prototype Evaluation				▨	▨			
6	Refine Requirements				▨	▨			
7	Engineer Product				▨	▨	▨		
8	Documentation				▨	▨	▨	▨	▨

CHAPTER 2
LITERATURE REVIEW

Chapter 2: Literature Review

In order to identify the technology, methodology and approach that will be used in developing Client/Server Game, literature survey has been carried out. In literature survey, relevant areas are discussed such as:

- Networks
- Client/Server Computing
- Network Communication
- Computer Games
- Programming Languages & Technologies

2.1 Networks

The number of computers in use worldwide, according to the International Data Corporation, is in excess of 100 million. Moreover, because of the expanding memory and processing power of these computers, users can put the machines to work on new kinds of applications and functions. Accordingly, the pressure from the users of these systems for ways to communicate among all these machine is irresistible. It is changing the way vendors think and the way in which automation products and services are sold. This demand for connectivity is manifested in two specific requirements: the need for communications software, and the need for network.

Why establish a computer network? That may seen obvious, but the reasons for doing so, as outlined next, shed light on what a network is and what it can do for an organization.

Program and File sharing

Networkable versions of many popular software packages are available at considerable cost savings compared with buying individually licensed copies. The program and its data files are stored on a file server and accessed by many network users.

Network Resource Sharing

Network resources include printers, plotters, and storage devices. The network provides a communication link that lets users share these devices.

Database Sharing

A database management system is an ideal application for a network. A network feature called record-locking lets multiple users simultaneously access a file without corrupting the data. Record-locking ensures that no two users edit the same record at the same time.

Economical Expansion of the PC Base

Network provides an economical way to expand the number of computers in an organization. You can attach to a network inexpensive diskless workstations that use the server's hard drive for booting and storage.

Workgroups

A network provides a way to create groups of users that are not necessarily located within the same department. Workgroups facilitate new "flat" corporate structures in which people from diverse and remote departments belong to special group projects.

Electronic Mail

Electronic Mail (e-mail) lets users easily communicate with one another. Messages are dropping in "Mailbox" for the recipients to read at a convenient time.

Groupware and Workflow software

Groupware and workflow software are designed specifically for networks and take advantage of e-mail systems to help users collaborate on projects, schedules, and document processing.

Centralized Management

A network provides a way to centralize servers and their data, along with other resources. Hardware upgrades, software backups, system maintenance, and system protection are much easier to handle when devices are located in one place.

Enhancement of the Corporate Structure

Networks can change the structure of an organization and the way it is managed. Users who work in a specific department and for a specific manager no longer need to be in the same physical area. Their office can be located in areas where their expertise is most needed. The network ties them to their department managers and peers. This arrangement is useful for special projects in which individuals from different departments, such as research, production, and marketing, need to work closely with each other.

2.2 Client-Server Computing

Ages ago (in Internet time), when mainframe dinosaurs roamed the Earth, a new approach to computer networking called "client/server" emerged. Client/server proved to be a more cost-effective way to build many types of networks, particularly PC-based LANs running end-user database applications. Many types of client/server systems remain popular today.

2.2.1 What Is Client/Server?

The most basic definition of client/server comes from the corresponding Usenet:

Client/server is a computational architecture that involves client processes requesting service from server processes.

In general, client/server maintains a distinction between processes and network devices. Usually a client computer and a server computer are two separate devices, each customized for their designed purpose. For example, a Web server will often contain large amounts of memory and disk space, whereas Web clients often include features to support the graphic user interface of the browser such as high-end video cards and large-screen displays.

Client/server networking, however, focuses primarily on the applications rather than the hardware. The same device may function as both client and server; for example, Web server hardware functions as both client and server when local browser sessions are run there. Likewise, a device that is a server at one moment can reverse roles and become a client to a different server (either for the same application or for a different application).

2.2.2 Components of Client/Server System

The Client

A networked information requester, usually a PC or workstation, that can query database and/or other information from a server.

The Server

A computer, usually a high-powered workstation, a minicomputer, or a mainframe, that houses information for manipulation by networked clients.

2.2.3 Logical layers of Client/Server System

Client/Server systems have three distinct logical layers, as illustrated below :

- Applications, user interface terminals, and data distributed across several remote machines.
- Middleware, which provide the interfaces or "glue" to enable remote machines to work together. Middleware hides details of the underlying systems software and hardware of particular computer platforms and network from the applications, so that applications can converse in standard ways across a network.
- The basic platforms, systems, and networking software and hardware, which support communications among the different parts of the PC.

2.2.4 Client/Server Pros and Cons

The pros

- Networked web of small, powerful machines

  If one machine goes down, your business stays up.

- Computer arrays with thousands of MIPS

  The system provides the power to get things done without monopolizing resources. End users are empowered to work locally.

- Some workstations are as powerful as mainframes but cost an order of magnitude less

  By giving you power for less money, the system offers you the flexibility to make other purchases or to increase your profits.

- Open Systems

  You can pick and choose hardware, software, and services from various vendors.

- Systems grow easily

  It is easy to modernize your systems as your needs change.

- Individual client operating environment

  You can mix and match computer platforms to suit the needs of individual departments and users

The Cons

- Maintenance nightmares

  Parts don't always work together. There are several possible culprits when something goes wrong.

- Support Tools lacking

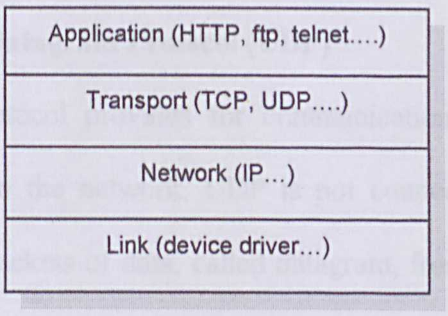
  With the client-server architecture, you must often locate or build support tools yourself.

- <u>Retraining required</u>

  The software development philosophy for the Mac or Windows is different from that of COBOL or C.

2.3 Network Communication

Computers running on the network communicate to each other using either the Transport Control Protocol (TCP) or the User Datagram Protocol (UDP)



When a programmer write a programs that communicate over the network, he or she are programming at the application layer. He do not need to concern with TCP and UDP layers.

2.3.1 Transport Control Protocol (TCP)

When two applications want to communicate to each other reliably, they established a connection and send data back and forth over the connection. This is analogous to making a telephone call. If person A want to speak to person B in Sabah, a connection is established when A dial B's phone number and B answers. Person A send data back and forth over the connection by speaking to one another over the phone lines. Like the phone company, TCP guarantees that data sent from one end of the connection actually gets to the other end and in the same order it was sent. Otherwise, an error is reported.

TCP provides a point-to-point channel for applications that require reliable communications. The Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP), and Telnet are all examples of applications that require a reliable communication channel. The order in which the data is sent and received over the network is critical to the success of these applications. When HTTP is used to read from a URL, the data must be received in the order in which it was sent. Otherwise, it will end up with a jumbled HTML file, a corrupt zip file, or some other invalid information.

2.3.2 User Datagram Protocol (UDP)

The UDP protocol provides for communication that is not guaranteed between two applications on the network. UDP is not connection-based like TCP. Rather, it sends independent packets of data, called datagram, from one application to another. The order of delivery is not important and is not guaranteed, and each message is independent of any other.

For many applications, the guarantee of reliability is critical to the success of the transfer of information from one end of the connection to the other. However, other forms of communication don't require such strict standards. In fact, they may be slowed down by the extra overhead or the reliable connection may invalidate the service altogether.

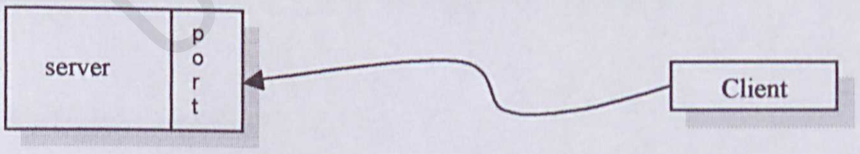
Consider, for example, a clock server that sends the current time to its client when requested to do so. If the client misses a packet, it doesn't really make sense to resend it because the time will be incorrect when the client receives it on the second try. If the client makes two requests and receives packets from the server out of order, it doesn't really matter because the client can figure out that the packets are out of order and make another request. The reliability of TCP is unnecessary in this instance because it causes performance degradation and may hinder the usefulness of the service.

2.3.3 Ports

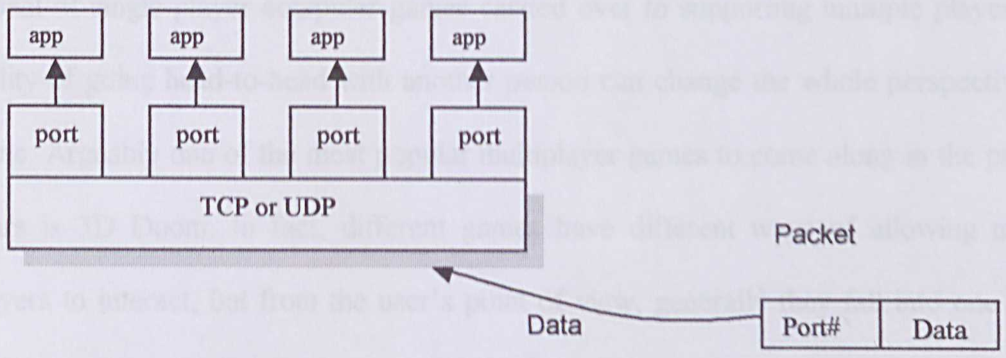
A computer has a single physical connection to the network. All data destined for a particular computer arrives through that connection. However, the data may be intended for different applications running on the computer. So, how does the computer know to which application to forward the data? Through the use of ports.

Data transmitted over the Internet is accompanied by addressing information that identifies the computer and the port for which it is identified. The computer is identified by its 32-bits IP address, which IP uses to deliver data to the right computer on the network. Ports are identified by a 16-bits number, which TCP and UDP use to deliver the data to the right application.

In connection-based communication such as TCP, a server application binds a socket to a specific port number. This has the effect of registering the server with the system to receive all data destined for that port. A client can then rendezvous with the server at the server's port.



In datagram-based communication such UDP, the datagram packet contains the port number of its destination and UDP routes the packet to the appropriate application.



Port numbers range from 0 to 65,535 because ports are represented by 16-bit numbers. The port numbers ranging from 0-1023 are restricted; they are reserved for use by well-known services such as HTTP and FTP and other system services. These ports are called well-known ports.

2.4 Computer Games

The Internet and World Wide Web have energized the already fast-moving world of computing and created previously unthinkable opportunities for communication among users. One of the interesting and attractive areas of application for the web is computer game. When games are networked on global scale, they offer a plethora of entertainment possibilities for users. Gaming on the web or network will truly change the way we all view entertainment – a new form of interactive entertainment.

2.4.1 Multiplayer games

Although AI (Artificial Intelligence) seems to be very important and has countless uses in games, it is hard to discount the human factor in multiple games. The appeal of multiplayer games in not hard to figure out. Considering how much people enjoy playing games together that do not involve computers, it really was a matter of time before the appeal of single-player computer games carried over to supporting multiple players. The reality of going head-to-head with another person can change the whole perspective of a game. Arguably one of the most popular multiplayer games to come along in the past few years is 3D Doom. In fact, different games have different ways of allowing multiple players to interact, but from the user's point of view, generally they fall into one of two categories : cooperative and competitive. Nevertheless, some games allow players the option of playing cooperatively or competitively.

<u>Cooperative Gaming</u>

In cooperative game, players work together toward some common goal. The old arcade game, Gauntlet, is a perfect example of this kind of game play. A good teamwork is needed in order to achieve the defined task.

<u>Competitive Gaming</u>

The other style of play is far more common. In a competitive game, players work against each other, and sometimes against the computer. Most board and card games are of this type, such as Bridge, Hearts and Othello.

2.4.2 Network Versus Non-Network Games

So far, the discussion of multiplayer games has avoided the issue of where multiple players' area physically located. From a programming perspective, there are two ways of handling a multiplayer game:

2.4.2.1 Non-Networked Games

A local game is the more common type. This is the kind of game where all of the players are at the same computer, and they take turns entering their moves. Such games usually use a joystick, because sharing keyboard maybe awkward.

Advantages:

- This type of game is easy to program.
- It requires only a machine in order to play.

Disadvantages:

- Usually, game play is slow, since each player must wait for all of the other players to compete their moves before he or she moves again. This can be especially aggravating if there are more than a few players.
- This type of games also presents difficulties for programmers, particularly in games that require each player to keep information secret from other players, for instance, Poker. The fact that all users share a common terminal is disadvantageous.

In light of recent advances in network standardization and programming libraries, however, this type of multiplayer games using the same machine is not compelling.

2.4.2.2 Remote/Network Games

Remote, or network games are multiplayer games that allow multiple players on different computers to compete across some kind of link, often a modem or network,:

Advantages:

- It allows physically distance players to interact.
- It also avoids the problem of secret information that local games suffer.

Disadvantages:

- Traditionally been more difficult to program than local game.
- Problem of speed, especially for real-time games. Each player needs constant updates about the position and status of other players. If this information is too slow in arriving, it makes the game unplayable.

However, networks of today have reached the point where sufficient data transfer rates can be attained for most of the rapidly paced games.

2.5 Programming Languages and Technologies

C++ and Java have become the programming languages of choice for writing software for operating systems, for computer networking, and for distributed client/server applications.

2.5.1 C++

C++ has evolved from C through a sequence of modifications to improve its imperative features and additions to support object-oriented programming.

The most essential part of support for object-oriented programming is the class/object mechanism. C++ provides a collection of predefined classes, along with the possibility of user-defined classes. The classes of C++ are data types, which can be instantiated any number of times. Such instantiations in C++ are merely object, or data, declarations. Class definitions specify data objects (called data members) and functions (called member functions). Classes can name one or more parent classes, providing inheritance and multiple inheritance, respectively. Classes inherit the data members and member functions of the parent class that are specified to be inheritable.

Operators in C++ can be overloaded, meaning the user can create operators for existing operators on user-defined types. C++ functions can also be overloaded, meaning the user can define more than one function with the same name, provided either the numbers or types of their parameters are different.

Dynamic binding in C++ is provided by virtual class function. These functions define type-dependent operations, using overloaded functions, within a collection of classes that are related through inheritance. A pointer to an object of class A can also point to objects of classes that inherit class A. When this pointer points to an overloaded virtual function, the function of the current type is chosen dynamically.

Both functions and classes can be templated, which means that they can be parameterized. For example, a function can be written as a templated function to allow it to have versions for a variety of parameter types. Classes enjoy the same flexibility.

2.5.2 Java

Java is based on C++ but was specifically designed to be smaller, simpler, and more reliable. Java has both types and classes. The primitive types are not objects based on classes. These included all of its scalar types, including those for integer, floating-point, Boolean, and character data. Objects are accessed through reference variables, but primitive type values are accessed exactly as the scalar values in purely imperative languages like C and Ada. Java arrays are instances of a predefined class, whereas C++

they are not, although many C++ users build wrapper classes for arrays to add features like index range checking, which is implicit in Java.

Java does not have pointers, but its reference types provide some of the capabilities of pointers. These references are used to point to class instances – in fact, that is the only way class instances can be referenced. While pointers and references may seem a great deal alike, there are some important semantic differences. Pointers point to memory locations, but references point at objects. This makes any kind of arithmetic on references nonsense, eliminating that error-prone practice. The distinction between a pointer's value and the value to which it points is the responsibility of the programmer in many languages, in which pointers sometimes must be explicitly dereferenced. References are always implicitly dereferenced, when necessary. So they behave more like ordinary scalar variables.

Java has a primitive Boolean type, used mainly for the control expressions of its control statements. Unlike C and C++, arithmetic expressions cannot be used for control expressions. Java has no record, union, or enumeration types.

One significant difference between Java and many of its contemporaries that support object-oriented programming, is that it is not possible to write stand-alone subprograms in Java. All Java subprograms are methods and are defined in classes. There in no construct in Java that is called a function or a subprogram. Furthermore, methods can only be called through a class or object.

Another important different between C++ and Java is that C++ supports multiple inheritance directly in its classes definitions. Java supports only single inheritance, although some of the benefits of multiple inheritances can be gained by using its interface construct.

Java includes a relatively simple form of concurrency control through its synchronize modifier, which can appear on methods and blocks. In either case, it causes a lock to be attached. The lock insures mutually exclusive access or execution. In Java it is relatively easy to create concurrent processes, which in Java are called threads. These threads can be started, suspended, resumed, and stopped, all with methods inherited from the parent class of all threads, Thread.

Java uses implicit storage deallocation for its heap-allocated objects (all Java class instances, or objects, are heap allocated), often called garbage collection. This frees the programmer from being concerned with putting storage back in the heap when it is no longer needed. Programs written in languages that require explicit deallocation often suffer from what it sometimes called memory leakage, which means that storage is allocated but never deallocated. This can obviously lead to eventual depletion of all available storage.

2.5.3 C++ Versus Java

Feature	Java	C++
Language	Pure object-oriented language.	Hybrid between procedural and object oriented.
Network abilities	Specifically attuned to network and Web processing	No relationship to networks or the Web

Allocating Memory	Programs must allocate memory to store information and perform calculations. When the program is through using the memory, it must release it back to the system so that it will be available to other parts of the system. In Java environments, there is a subsystem called a *garbage collector* that detects when a program no longer needs a piece of memory, and consequently releases it back to the system	C++ programs must explicitly release memory back to the system. One major cause of bugs in C++ programs is that programmers forget to explicitly release memory back to the system. This memory is "leaked" and will not be available until the program terminates.
Parallel processing	Supports multithreading.	No multithreading.
Execution Time	The executable files (called *class files*) produced by a Java compiler contain collections of platform-independent bytecode which cannot be run on a target platform without translation into binary instructions suitable for each target platform's CPU. The JVM is responsible for performing this translation. Bytecode interpreters perform many times slower than comparable C++ programs because each bytecode instruction must be interpreted every time it is executed, which can lead to a great deal of unnecessary overhead.	Fast execution.

2.5.4 Active X

Microsoft's Active/X™ technology is not a language but a means of deploying small executable components that can be run within the confines of a client's web browser. It is derived from their OLE and COM strategy of the past few years. Usually these components are compiled from C++ source and distributed as a true executable. This gives rise to the following points:

- Because it contains target machine code the component is not portable and can only run on a client machine for which it has been built; for the moment this implies the client must be running Win32. Also, since machine code is inevitably larger than a bytecode format, this implies that download times of Active/X components may be longer.
- Active/X components are executed raw on the target machine without the safety buffer of a virtual machine to filter out rogue requests. This makes for a potential security risk but allows the applets to be much more powerful than their Java counterparts. Microsoft's answer to the security issue is to *digitally sign* each Active/X component so that it is possible for the client to verify that they have been received unmodified from a known and *trusted* supplier.
- The ability to execute Active/X components is built-in to the Microsoft Internet Explorer. Other browsers can install a plug-in to allow them to benefit from web pages with these embedded components.

2.5.5 Java Applets

Java is an architecture neutral and portable object-oriented language derived from C++. It can be used either to build applets for execution inside web browsers or applications capable of supporting themselves. Most of the excitement surrounding Java at the moment is aimed at the former ability.

Java applet classess are compiled to a compact bytecode format that is independent of any target machine and these class files are distributed across the Web alongside the HTML pages that include them. The bytecodes are then interpreted by a Java virtual machine (VM) that is (usually) built in to the client web browser. Sometimes the bytecodes are dynamically compiled at the client side to aid performance. The existence of the VM interpreter at the client side gives rise to two important features of Java:

- The VM implementation can be different for different client side hardware. This allows Java applets to execute on a wide range of target systems from Windows to Macs to Unix. It inevitably means, though, that Java applets cannot make use of useful operating system features that are not common across all target environments.
- Since all bytecode execution is in some way interpreted by the VM this allows for a secure situation in which rogue applets can be prevented from damaging the target machine. This so-called *sand-box* approach encourages end-user confidence in the distribution mechanism but it does significantly restrict the possibilities of what can legitimately be done by a Java applet.

2.5.6 Active X Controls versus Java Applets

Feature	Java	Active X
Safety - Protection from viruses or malicious code downloaded automatically across the Internet	The inbuilt Java security manager is consulted every time a Java application tries to access the disk, the network ports and external processes. Within a browser, the security manager will prohibit most types of access and so a Java applet downloaded off the Internet cannot cause any damage. Unfortunately, the security manager prevents an applet doing a lot of useful things too, so in time it is likely that the security manager will be weakened. As with all security, though, there are known loopholes... The new Java 1.1 from Sun allows Java applets to be code signed in much the same way as ActiveX controls. Then the Java security manager may be weakened.	ActiveX controls attempt to be safe by being "digitally signed" with an authentication code identifying the manufacturer. This allows a user to choose to download a control written by the well known and trusted Microsoft but not an equally legitimate control written by the lesser known Spiral Software. The ActiveX author is required to get a certificate to enable signing of controls. Apart from the issue of which manufacturers to trust, there is the danger that an unscrupulous web page designer could exploit weaknesses in a legitimate control.

Speed on a standard PC	Java is currently slow. This is a disaster for animations. Of course, Java should be a lot faster on Sun's JavaStation or any other system designed to run Java. Faster "Just-in-time" Java compilers from Microsoft and Netscape help to speed Java up a little.	
Existing controls or applets		There are many ActiveX controls now available for Windows95/NT (see the Microsoft ActiveX gallery). This includes many previous OLE and .ocx controls (but not all). There are very few ActiveX controls available for Windows 3.x and other operating systems. Note that .vbx controls cannot easily be converted into ActiveX controls.
Portability	Java will run unmodified on any computer which supports the Java virtual machine. Java VMs are widespread and also available in the latest Internet browsers from Microsoft and Netscape	ActiveX controls have to be specially written for each machine and operating system

Ease of programming	Java is an elegant language but still somewhat specialized and requiring professional programmers.	ActiveX controls have traditionally been written in C/C++ - popular, but specialized, languages. Whilst the ActiveX interface is quite complicated the task is eased somewhat by Microsoft base C++ classes (part of the MFC library). However, Microsoft's Visual Basic 5 (Control Creation Edition) will allow ActiveX controls to be produced very easily, but only for the Windows95/NT operating system. Microsoft have no plans to allow Visual Basic to make controls for Windows 3.x or other operating systems.

CHAPTER 3
SYSTEM ANALYSIS

Chapter 3: System Analysis

3.1 Define and analyze functional requirements

A Functional requirement is a description of activities and services a system must provide. It is frequently identified in terms of inputs, outputs, processes, and stored data that are needed to satisfy the system improvement objectives. The functional requirements for the system are summarized as stated:

Process Network Game Play

The system should be incorporated into the network and users are able to access this game system by opening up a browser connection to the specific game server. Upon logging on to the system, users will then be served by a game play with another user connected to the network.

Storing Score

The system should perform a score storing system for users to record their best scores. After a game is over, users can select to input their alias and email reference and the system will automatically store these details together with the game score and date of the play made. All the data will be updated periodically and stored in descending order.

Multiple User Request

The system will not only permit the first group of players who connect to play the game. It should be able to process multiple game requests at once.

Viewing High Score

This function should allow users to check out the current score board, which lists the details of the top 5 record holders.

Sending Messages

The system should be accompanied by a service available for users to communicate with each other during a match. The users will be able to send their messages to the entire room or to a targeted user. The system will receive and process messages received and then broadcast it to each user.

Information Display

The system will guide the users by displaying meaningful information necessary to assist them on how to start and respond towards the system.

Logoff

The logoff option should allow users to logoff from the system if they are logged in.

Rules

This section should provide users with information of the rules of monopoly.

3.2 Define and analyze non-Functional Requirements

Non-functional requirements are essential definition of system properties and constraints under which a system must operate. The non-functional requirements for this system are summarized as stated:

<u>Response Time in a Game Play</u>

Response time from the game server and also from the database is a crucial issue especially in playing games via the network and in establishing a connection to the database. When users send a move, message or request during the a game, they should not be kept waiting for a long time for the results.

<u>Immediate Updates</u>

All the state of the game on both players' system should be notified and updated exactly as soon as there is a move made by players involved. Any information loss will lead to confusing results and user will lost interest in the system.

<u>User Friendliness</u>

The design of the system and its interface should be user friendly and easily understood by all levels of network users. Generally, the design of all the interfaces should conform to the following criterions:

- Consistent, in terms of screen design and error messages displayed.
- Offers a high degree of understandability and avoid memorization of events and commands.

Multiple Platform Support

The game system should be able to facilitate user access from a great variety of browsers and computer platforms. This is to allow as many users to make use of the system, thus, increasing the distribution of the game over the network.

Reliability, Accuracy and Robustness

The system should be able to perform accurately the message sending functions as requested by the users. Also, the game server should be as robust as possible in processing multiple connections from different users. Most importantly, the system is able to handle unexpected error and echo back with proper responses, thus, enhancing reliability and robustness.

3.3 Development Tools chosen

Based on the literature survey on the programming languages in the previous chapter, Java is chosen to be the development programming language based on the following reasons:

Portability

Java is a portable language and the programs written in Java can run on many different computers. Java programs are compiled into machine-independent bytecodes. they run consistently on any Java platform.

Shorter Development Time

Development time may be as much as twice as fast versus writing the same program in C++.

Multithreading

Most of today's programming languages, including C and C++, do not include features for expressing parallel operations. Java includes capabilities to enable multithreaded applications. (i.e., applications that can specify that multiple activities are to occur in parallel).

Database

Java provides the ability to use JDBC (Java Database Connectivity) to connect to a Microsoft ODBC (Open Database Connectivity) data source through the JDBC-to-ODBC bridge. This is essential in retrieving the scoring information from the database.

Networking

Java provides a rich complement of networking capabilities and will likely be used as an implementation vehicle in computer networking. In fact, Java-compatible browsers use this ability of the Java platform to the extreme to transport and run applets over the Internet. Java Remote Method Invocation (RMI) enables Java programs to communicate with each other via method calls that are automatically sent across the network.

Graphics

Java contains many sophisticated drawing capabilities as part of the Java2D API and Java3D API. Java2D API enables developers to easily incorporate high-quality 2D graphics, text, and images in applications and in applets. Java3D API is an interface for writing program to display and interact with three-dimensional graphics. It provides the functions for creation of imagery, visualizations, animations, and interactive 3D graphics application programs.

Multimedia

The packages of classes that are an integral part of the Java programming world provides extensive multimedia facilities that will enable programmers to start developing powerful multimedia applications immediately. Java Media Framework provides enhanced processing of images and enhanced audio playback supporting many of today's popular audio formats. The JMF also includes video playback capabilities for several video formats.

3.4 Define system requirements

The system requirements are defined in two points of view, the development environment and the run time environment.

3.4.1 Development Environment

Hardware Requirements

The recommend hardware requirements for the development environment are listed as the following:

- Pentium ® 166 or equivalent
- 16 MB RAM
- 32MB available hard drive space
- 4xCD-ROM
- SVGA Graphics Adapter
- Keyboard & Mouse

Software Requirements

The recommend software requirements for the development environment are listed as the following:

- Window ® 2000, ME, XP, 95, 98
- Java Development Kit 1.3.1
- Microsoft FrontPage 2000
- Internet Explorer, Netscape Communicator

3.4.2 Run Time Environment

Hardware Requirements

The recommend hardware requirements for the run time environment are listed as the following:

- Pentium ® 166 or equivalent
- 16MB RAM
- 32MB available hard drive space
- 4xCD-ROM
- SVGA Graphics Adapter
- Keyboard & Mouse

Software Requirement

The recommend hardware requirements for the run time environment are listed as the following:

- Window ® 2000, ME, XP, 95, 98
- Java Runtime Environment 1.3.1

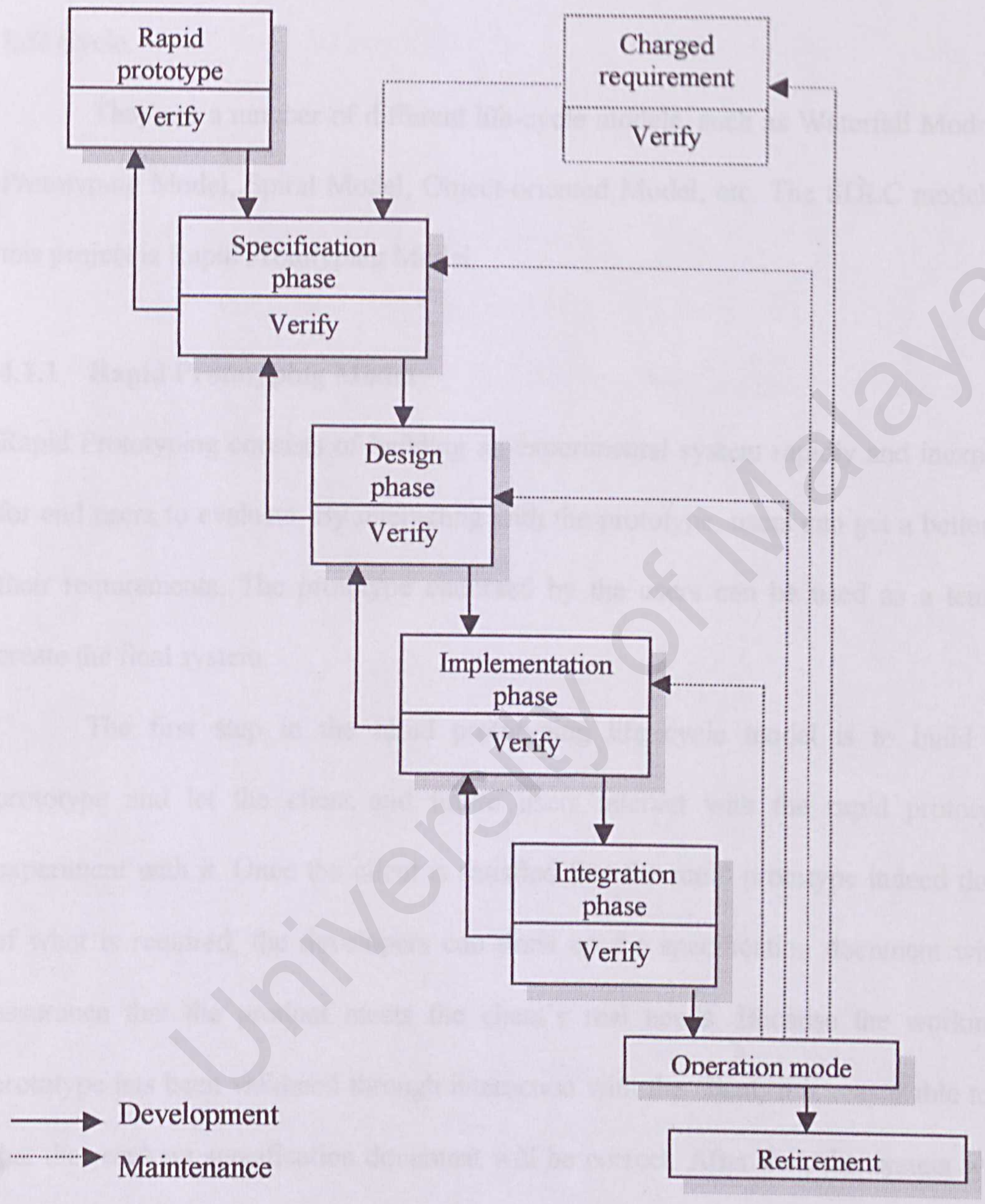
- Internet Explorer, Netscape Communicator

CHAPTER 4
SYSTEM DESIGN

CHAPTER 4
SYSTEM DESIGN

Chapter 4: System Design

4.1 Software Development Life Cycle (SDLC)



Rapid Prototyping Model

A system must goes through a series of development phases before it can be implemented. Typically, the system is specified, designed and then implemented. The series of steps through which the product progresses is called the Software Development Life Cycle.

They are a number of different life-cycle models, such as Waterfall Model, Rapid Prototyping Model, Spiral Model, Object-oriented Model, etc. The SDLC model used in this project is Rapid Prototyping Model.

4.1.1 Rapid Prototyping Model

Rapid Prototyping consists of building an experimental system rapidly and inexpensively for end users to evaluate. By interacting with the prototype, users can get a better idea of their requirements. The prototype endorsed by the users can be used as a template to create the final system.

The first step in the rapid prototyping life cycle model is to build a rapid prototype and let the client and future users interact with the rapid prototype and experiment with it. Once the client is satisfied that the rapid prototype indeed does most of what is required, the developers can draw up the specification document with some assurance that the product meets the client's real needs. Because the working rapid prototype has been validated through interaction with the client, it is reasonable to expect that the resulting specification document will be correct. After that, the system will go to the design phase.

In the design phase, the system is constructed. Even though rapid prototype has been hurriedly assembled, the developers can confirm precisely the users' requirements from it. Upon completion of this phase, the system is implemented and integrated. The system is tested to ensure that no flaws or bugs exist.

Once the system has been accepted, maintenance begins. Depending on the maintenance that has to be performed, the cycle is reentered either at the requirement, specification, design, or implementation phase.

4.1.2 Advantages of Rapid Prototyping Model

- Changes can be made earlier in the development stage.
- System or elements that is not in working condition can be eliminated.
- User's need and expectation can be followed more closely.
- Misunderstanding between software developers and users may be identified as the system functions are demonstrated.
- Missing user services may be detected.
- Difficult-to-user or confusing user services may be identified and refined.
- The prototype serves as a basic of writing the specification for a production quality system.

4.2 System Design

The system is designed using the client/server technique. In fact, the design of the system can be divided cleanly into a client side and a server side. These two components are logically separated, communicating entirely through a game protocol defined between them.

4.2.1 The Server

In any game, the server side of the game acts almost like a referee, managing the different players and helping them communicate effectively. More specifically, a game server takes on the role of handling the network connection for each player, along with querying for and responding to events for the players. The role of a generic game server can be broken down into the following actions:

- Initialize the server socket
- Wait for a client to connect
- Accept the client connection
- Create a daemon thread to support the client
- Go back to step 2.

Because Monopoly is a multiple player game, the first job of the server is to group at least two players (clients) as they connect to the game. A more limited approach would be to permit only the first group of players who connect to play the game. But users are more thorough than that and demand a more powerful game server. The game server enables multiple games to be played at once, simply by grouping additional client players together for each game. In this way, a typical game server session might have six or eight groups playing at once. Of course, the players know only about the other player in their immediate game.

After the server has detected at least two players and groups them for a game, it becomes the responsibility of the server's daemon thread to dictate the flow of the game between the players. The daemon accomplishes this by informing each player of the state of the game, while also modifying the state according to each player's turn. The responsibilities of the game server and daemon can be summarized as follows:

- Accept client player connections
- Group players to form separate games
- Manage the flow of the game
- Communicate each player's move to the other players.
- Notify the players of the state of the game

4.2.2 The Client

The client portion of a network game corresponds to the application being run by each player. Because game players interact with the client, the client program is usually much fancier than the server in regard to how information is displayed. As a matter of fact, game servers typically do not even have user interfaces; they crank away entirely behind-the-scenes doing all the dirty work while the client applets dazzle the users.

The basic responsibility of a game client is to connect to the server and communicate the user's actions, along with receiving game state information from the server and updating itself accordingly. Of course, along with this comes the responsibility of displaying the game graphics and managing the entire game interface for the user. The following is a summary of what network functionality the game client needs to provide:

- Connect to the server
- Notify the player of the connection and game state
- Communicate the player's move to the server
- Receive the other player's move from the server
- Update the game with the state received from the server

4.2.3 Running The Game

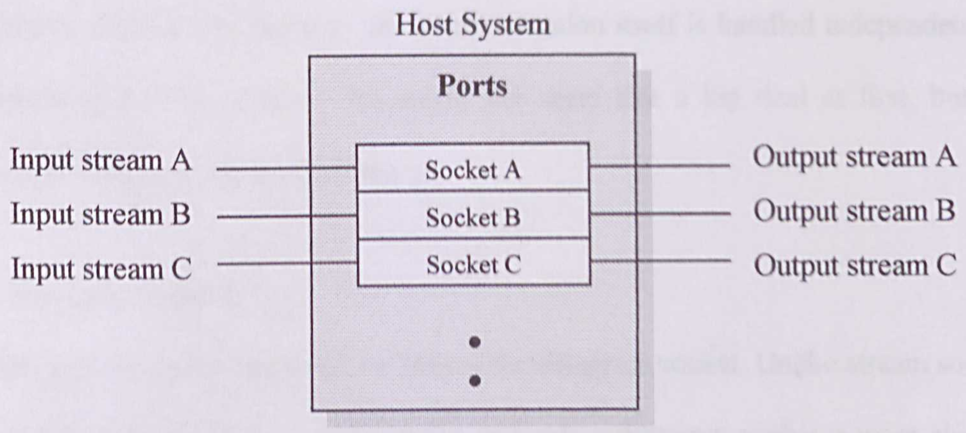
For a network game to work, the game server must always run in the background. It must somehow be launched by the server or by some type of system startup feature making it available to connect clients requesting to play the game.

When a game client shows up to play a game, the game server accepts the client's connection request and takes on the role of hooking the client up with others client to play a game. The game server is entirely responsible for detecting when clients arrive as well as when they leave, creating and canceling game sessions along the way. Because the game server is being run in the background all the time, it must be extremely robust. Also, since the game server is responsible for detecting and grouping clients, it is imperative that the server be running at all times.

The game server is the core of the game and must be running for the clients to work. To get the game running, the server must run first. The other half of network game is the client, which is an application that runs on others computer that connected to the server. After the server is up and running, start the client application. After running the game client application, the users should see the game interface.

4.3 Game Communication

Java network game programming uses a particular type of network communication known as sockets. A socket is a software abstraction for an input or output communication medium. Java performs all its low-level network communication through sockets. Logically, sockets are one step lower than ports; Sockets can be use to communicate through a particular port. So a socket is a communication channel that enables the transfer of data through a certain port.



Data can be transferred through multiple sockets for a single port. Java provides socket classes to make programming with sockets much easier. Java sockets are broken down into two types: stream sockets and datagram sockets.

4.3.1 Stream socket

A stream socket, or connected socket, is a socket over which data can be transmitted continuously. By continuously, it not necessarily means that data is being sent all the time, but that the socket itself is active and ready for communication all the time. Stream socket can be thought of as a dedicated network connection in which a communication medium is always available for use. The benefit of using a stream socket is that information can be sent with less worry about when it will arrive at its destination. Because the communication link is always "live", data is generally transmitted immediately after a user sends it.

Java supports stream socket programming primarily through two classes: Socket and ServerSocket. Socket provides the necessary overhead to facilitate a stream socket client, and ServerSocket provides the core functionality for a server. Most of the actual code facilitating communication via sockets is handled through input and output streams connected to a socket. In this way, the communication itself is handled independently of the network socket connection. This might not seem like a big deal at first, but it is crucial in the design of the socket classes.

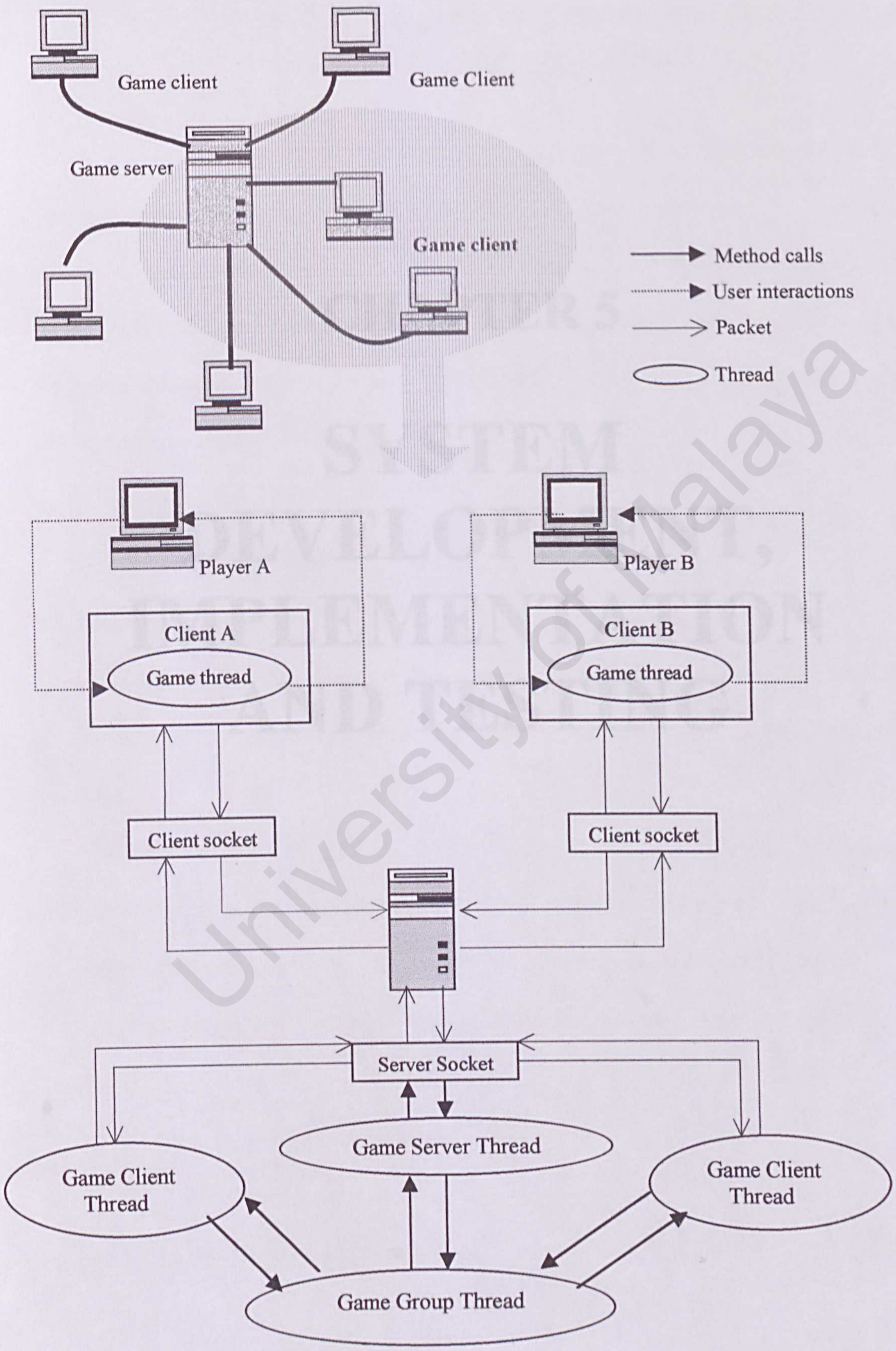
4.3.2 Datagram socket

The other type of socket supported by Java is the datagram socket. Unlike stream sockets, in which the communication is akin to a live network, a datagram socket is more akin to a dial-up network, in which the communication link isn't continuously active. A datagram socket is a socket over which data is bundled into packets and sent without requiring a live connection to the destination computer.

Because of the nature of the communication medium involved, datagram sockets aren't guaranteed to transmit information at a particular time or even in any particular order. The reason datagram sockets perform this way is that they don't require an actual connection to another computer; the address of the target computer is just bundled with the information being sent. This bundle is then sent out over the Internet, leaving the sender to hope for the best.

On the receiving end, the bundles of information can be received in any order and at any time. For this reason, datagrams also include a sequence number that specifies which piece of the puzzle each bundle corresponds to. The receiver waits to receive the entire sequence, then puts them back together to form a complete information transfer. Datagram sockets are less than ideal for network game programming because of the implied time delays, heightened reliability issues, and sequencing complexities.

4.4 Network Monopoly Model



CHAPTER 5

SYSTEM DEVELOPMENT, IMPLEMENTATION AND TESTING

Chapter 5: System Development, Implementation and Testing

5.1 Introduction

This chapter describes the processes and strategies of transforming the system into workable models and programming codes (development), setting up the system for actual use (implementation), and testing the system for usability, reliability and performance (testing). These phases are the final phases of the System Development Life Cycle and will reflect upon the design of the system. A good design will produce a good system and vice versa.

5.2 System Development

5.2.1 Introduction

In the system development, the design has to be transformed into workable models of the system and programming codes must be written for the processing involved.

This is not a task to be taken lightly, for any mistakes or errors in developing or programming the modules will have consequences on the system quality and integrity, especially early mistakes that will corrupt all related development that come after.

To reduce errors and increase productivity and system quality, the following are needed:

1. System Development Strategy
2. System Development Platform
3. System Development Techniques

5.2.2 System Development Strategy

A modular approach is taken to the system development process. A module is an independent portion of the system that does a separate function of the system and is loosely related to others modules. When working together, the modules make up the complete system.

The modules are already identified in the design phase, namely:

1. Client Modules
   - 1.1 Game Modules
   - 1.2 Chat Modules
2. Server Modules

Each module has sub components of its own and can be developed relatively independent of others modules. The steps are:

1. Develop a server module that provides services to client
2. Develop a client module
3. Develop connection between client and server
4. Test and integrate both modules
5. Develop client's sub modules
6. Test and integrate the modules
7. Repeat step 4 until all sub modules are completed
8. System level integration
9. System implementation
10. System testing
11. System evaluation

Through this strategy, it is hoped that the number of errors can be reduced and a more reliable and usable system can be produced.

5.2.3 System Development Platform

A system development has to be chosen carefully to produce a realistic system. It is recommended highly that the system be developed in an environment identical to the environment it is destined. However this is not always practical because of the exorbitant cost of software and hardware, unavailability of software or hardware, development difficulties and so on. Therefore, this system is developed in a scaled down environment of the actual environment of its service.

5.2.3.1 Hardware Configuration for Development

For system development, a single PC is used to act as a server. The basic hardware requirements for this PC are:

1. Color monitor
2. Mouse
3. Keyboard
4. Sound card
5. Pentium 166 processor or better
6. Memory 32MB
7. Hard Drive with at least 10MB of free space

However, these are only the bare minimum of requirements and offer poor performance, a more powerful PC is highly recommended. Notice that Network Card is not including in the above list, this is because in development process, a same PC is used to act as Server and also Client. The connection is achieved through IP 127.0.0.1.

The actual hardware platform in which the system is developed is as below:

- 15 inch color monitor
- Serial Mouse
- 104 Keyboard
- Pentium II 266 Processor or better.
- Memory 64 MB
- SVGA with 256 Colors
- Network Card and connection to LAN

5.2.3.2 Software Configuration for Development

The software configurations are described in the table below. This is a tone-down environment of the environment of its actual services.

Software	**Description**
Microsoft Window ME	Server & Client Operation System
Notepad	Coding
Internet Explorer Version 5.5	Viewing the applet client
Paint	Image Manipulation
Java Development Kit 1.3.1	Compile & Run the code
Microsoft FrontPage	Web page design and coding

5.2.4 System Development Techniques

System development techniques are the practices and knowledge that are applied in the actual development:

1. Server System Development
2. Client System Development
3. Web Page Development
4. Prototyping
5. Debugging

5.2.4.1 Server System Development

The whole system that developed is a networked system. Server Application is the core of the system. It is used to manage the connection between all client computers. The server application is developed using Java. It is designed to support up to 20 games at a single time, support nearly 80 users.

When the coding is done, the application is tested. Not the whole application can be tested. Some of the functions cannot be tested until the client application is developed. Those functions are mostly related to Client-Server connection, messages passing, etc. The cycle is continued until no mistake is detected.

5.2.4.2 Client System Development

The client system is the system that interfaces to users. It consists of all the elements that users needed to play the game, including game board, dices, chat window, etc.

The Client is designed using Java. When the interface has been developed, more control codes and functions are added to it for enhanced programming capabilities. Two types of clients are developed: one is developed using Java Application while the other is using Java Applet.

The Client Application is the default system that applied in a Local Area Network. The application is installed on the user's machine and has complete access to all system resources. Client applet, however, is developed for future enhancement as web-based game. The applet is resides in the web server, user must use the Web browser, like Internet Explorer or Netscape Communicator to load all the classes' files from the server before start to play the game.

When the coding is done, the system is then integrated with the server system and tested. If any mistakes are detected, changes are immediately made and document saved. The cycle is continued until no mistake is detected.

5.2.4.3 Web Page Development

A web page is needed in order to load the Java Applet. The main tool used in web page development is Microsoft FrontPage.

The pages are designed using Microsoft FrontPage. When the page is done, it is viewed through Internet Explorer. If any mistakes are detected, changes to the page are immediately made and the document saved. Then the page is refreshed in browser. This cycle is continued until no mistake is detected.

5.2.4.4 Prototyping

Because of time constraint, prototyping technique is not used extensively in system development.

Only the user interface for some modules is prototyped. These prototypes are demonstrated to friends for advice. The feedback is used to improve the subsequent system and the prototypes are discarded.

5.2.4.5 Debugging

Many programmers consider debugging as the most boring and unrewarding part of their work. However, no matter how careful or proficient a programmer is, mistakes do creep into the codes. Therefore it is very important to learn how to debug effectively in order to increase program quality and programmers' productivity.

Due to the coding is developed using notepad, a non-Java Language Editor, the debugging process is fully depended on the error messages displayed in DOS prompt when compiling the codes using Java Development Kit (JDK).

When handling logical error, it is important to insert temporary debugging codes at certain intervals to track movement of the program and return values of key variables in strategic location inside the program. This works almost as good as break points in a Visual Basic Debugger. Sometimes it is just as important to identify the codes that are working instead of those which are not.

Through these methods are rather primitive, they are still very useful at a pinch.

5.3 System Implementation

5.3.1 Introduction

System implementation is the process of setting up the system for actual services. The strategy is a straightforward deployment.

5.3.2 Deployment

The deployment of Network Monopoly is not complicated. It consists only of a few steps. The deployment stated here is for client application, not the client applet.

When the system is being implemented, a single PC must be chosen as server computer to manage all the client connection. The server's files need to be copied to the desired folder of the server computer. The client files then need to be copied to any computer in the network.

The detailed steps of deployment can be found in the User Manual that is reproduced in Appendix C.

5.3.3 Game Instruction Briefing

Since the monopoly game is the most popular game in the world and most players are likely to be familiar with it, the game instruction briefing is expected to be easy. However, a comprehensive user manual has been prepared which contains guidance and instructions to use game system. In addition, game instructions are also printed out in the user interface along the game play and a complete help files are also provided. This certainly should guide the users well. A copy of the user manual is reproduced in Appendix C.

5.4 System Testing

5.4.1 Introduction

Testing of a system does not actually come at the end of the system development but should be carried out rigorously throughout the development phase. Testing is important in any system as it ensures that the system functions properly and conforms to the requirements specified.

There are three stages of testing involved in Network Monopoly:

1. Unit Testing
2. Integration Testing
3. System Testing

5.4.2 Unit Testing

In the unit testing, the most basic units of the system – the individual modules are tested. A module is tested independent of others modules. It is followed by the testing of the relation between modules and game data integrity.

The main objective of unit testing is to ensure program accuracy, data integrity, usability, reliability, and efficiency at the module level.

5.4.3 Integration Testing

Integration testing is a systematic approach for constructing the application while conducting tests to uncover errors associated with interfacing of different parts or modules. The objective is to ensure that the different unit-tested modules in Network Monopoly can function smoothly together to the exaction of the system requirements.

The major concerns here are the game data integrity. It is important to make sure that all game users receive the same data during the game play.

5.4.4 System Testing

After all the modules are completed, the entire system must now be validated. The system must then be compiled with others system elements such as hardware, and others computer systems. The objective of system testing is to verify the functional and non-functional requirements of the system. The functional and non-functional requirements for Network Monopoly as defined in chapter 3.

CHAPTER 6

SYSTEM EVALUATION

Chapter 6: System Evaluation

6.1 Introduction

As the final phase in the life cycle approach, this chapter evaluates the system to identity its strengths and limitations. As suggestion to further improve this system, the possibility to enhance the system is also explored.

6.2 Strengths of Network Monopoly

As the product of months of analysis and design, and also weeks of development and debugging, the system has several strengths.

1. User Friendly Interface

The interface of the system is simple and easy to use. Users can use mouse to point and click their way to play the game. Clear, precise instructions and guidance messages are provided during the game play. The user can follow this guidance then the chance an error is made is very low.

2. Multiple Platform Support

The system is developed using Java. Java is a machine-independent language. It is able to run in many different computers. This consideration has resulted in the system that is supportable by multiple platforms and browsers. Thus, increasing the distribution of the game over the network.

3. Multiple Game Play Support

The system will not only permit one game play at a time. The server application that developed is able to process multiple game requests at once. It is able to support up to 20 games at a time, support nearly 80 users at once.

4. Features "Chat" Ability

The system will not only permit the users who connect to play the game. It also features a "Chat" window that allows users to communicate with each other during the game play. The users will be able to "say" things to others users through this window. This leads the way to generate a strong level of interaction among users.

5. Dynamic Game Play

The system that developed is able to serve players with more dynamic game play. It allows player to logoff from the game whenever they wish to. They are allowed to quit from the game although they are not finish in playing the game. The system also will provides an interface that display all the games that currently available before a player want to join with the game. Player is able to choose to join in any game between all the games available.

6.3 Limitations of Network Monopoly

Despite the best intentions and designing efforts, the system still has several limitations. This is due mainly to inexperience of the developer, limitation of the tools, time constraint and so on.

1. Lack of Scores Storing System

The system does not provide scores storing capability. The users are not able to store their own score for future references. The lack of scores storing system makes it difficult for users to compare their scores with others.

2. Limitation on Players Support

Currently, only 2 to 4 players are supported for a single game. For official monopoly game, until 8 players is allowed to play in a single game. The system also can support until 20 games in a single time only, this limitation has make it not adequate if applied in a large network or converted as web-based gaming system.

3. Lack of Timely Alert

No timely alert feature is available in the system. It is required for all players to always makes their game moves faster. If a player forgot to do so, others players will also kept in waiting for long time. This will slow down the game play progress.

6.4 Future Enhancement

The following reflects the potential improvement to the system. The following list also gives a few suggestions for future evolution of the system, which are not covered in the original scope.

1. Adding More Players

Currently only 1 to 4 players has been allowed to play in a single game. More players can be added in future. For monopoly game, until 8 players are allowed to play in a single game.

2. Adding High Scores Board

The system can add a score storing system for users to record their best scores with their alias and email reference. This feature can be done by the uses of database or file, to store the high score records. The records then are able to be retrieve and display it to the users.

3. Provide More Interactive Game Play

A more interactive game play can be developed to provide more interactive features to the users such as animations, music, graphics or images.

4. Add the Timely Alert Features

Add a feature that send alert to a player to remind him or her of making game moves if he or she did not make moves in a duration of time. This is important in a game play because if a player did not making her/his move very long time, others players will also kept in waiting for a long time.

CHAPTER 7

CONCLUSION

Chapter 7: Conclusion

7.1 Introduction

The final chapter highlights some of the problems faced throughout the project duration. These problems include lack of development experience, choice of development tools, difficulty in obtaining licensed software.

7.2 Problems and Solution

1. Lack of Development Experience

The developer has had no prior experience or knowledge in developing network game system. This lack of experience and knowledge has proved to be an obstacle in the beginning. The developer has struggled to understand the concepts of network programming and application and differentiate them from the conventional programming concepts to which he is more accustomed.

Solution:

Luckily, there are abundant reference materials available for the developer on the subject. This is especially true for Java. A lot of Java tutorials are freely available on the web. Further more, there are many Java reference books in the Library. Therefore, this problem only proved to be a slight delay in the schedule with several weeks with little progress. After the knowledge and skills has been familiarized, everything went on rather smoothly.

2. Difficulty in obtaining licensed software

The system is mostly developed outside of the faculty labs. The major problem to work at home is the difficulty in obtaining the software needed for the project. The exorbitant price of the software is a major obstacle.

Solution:

The solution is to use a tone-down environment to develop the application. Instead of using more advanced Java editor like Microsoft Visual J++ or Borland JBuilder, notepad and Microsoft FrontPage are used to develop the pages and Java applet. Paint is used in image manipulation instead of using more sophisticated software like Adobe Photoshop or CorelDraw. Luckily, Java Development Kit (JDK), the Java Interpreter, is freely downloaded from the web.

3. Difficulty in Error Checking and Recovery

All the coding of the system is developed using notepad, a non-JAVA Language Editor. The Error checking and recovery process becomes difficult because the process is fully depended on the error messages appear while compiling the codes in DOS prompt using Java Development Kit. In addition, the error messages that appear are too general for recovery process.

Solution:

The solution is to insert temporary debugging codes at certain intervals to track movement of the program and return values of key variables in strategic location inside the program. This works almost as good as break points in a Visual Basic Debugger.

7.3 Project Conclusion

The network monopoly game system is designed and developed for the use in the Local Area Network. The core function of the project is about creating a network monopoly gaming system to play within the network. The second major feature of the system is the chat window that allows users of the system to communicate with each other during the game play. Overall, the final product has fulfilled almost all the basic functional and non-functional requirements as specified in the design.

However, as common for any project, there is still room for improvement for the system. The interface of network monopoly, though adequate and simple to use, can still use much dressing up to look more attractive. The lack of scores storing system makes it difficult for users to compare their scores with others.

As the project is developed using two Java technologies: Java application and Java applet, it can easily convert to serve as web-based gaming system. Although the system is able to process multiple game requests at once, but only for maximum 20 games only. This limitation causes it not so adequate to serve as web-based system. The basic concepts and structure of the system still applicable to web-based system, but the system cannot be readily converted without major redesign and recoding.

Developing Network Monopoly is a very novel and exciting experience. On one hand, the system explores the cutting edge information Technology and on another it offer practical experience in designing and developing a system for real usage. In such project, the developer will surely encounter many novel concepts and theory especially if he has no prior network game development experience. The knowledge, skills and experience gained in designing and developing this system will be of great use in the future.

REFERENCES

References

- Thomas Petchel. Java 2 Game Programming, Prima Publishing. 2001
- Wayne Holder & Doug Bell. Java Game Programming For Dummies®, Hungry Minds, Inc. 1998
- Chris Crawford : The Art of Computer Game Design, Washington State University. 1997
- Mary Campione & Kathy Walrath. The Java Tutorial : Object-oriented programming for the internet, Addison-Wesley. 1998
- H. M. Deitel & P. J. Deitel. Java How to Program, Prentice Hall. 1999
- Neal Ford, Ed Weber, Talal Azzouka, Terry Dietzler, Jennifer streeter & Caser Williams. Borland Jbuilder 3, Sams Publishing. 1999
- William Stallings & Richard Van Slyke. Business Data Communications, Prentice Hall. 1998
- Behrouz A. Forouzan. Data Communications and networking, McGraw Hill. 2000

APPENDIX

Appendix A

The official Rules of Monopoly

Object

The object of the game is to become the wealthiest player through buying, renting and selling property.

Equipment

The equipment consists of a board, 2 dice, tokens, 32 houses and 12 hotels. There are Chance and Community Chest cards, a Title Deed card for each property and play money.

Preparation

Each player chooses one token to represent him/her while traveling around the board. Each player is given $1500 divided as follows : 2 each of $500's, $100's and $50's; 6 $20's; 5 each of $10's, $5's and $1's. All remaining money and other equipment go to the Bank.

Banker

Select as Banker a player who will also make a good Auctioneer. A Banker who plays in the game must keep his/her personal funds separate from those of the Bank. When more than five persons play, the Banker may elect to act only as Banker and Auctioneer.

The Bank

Besides the Bank's money, the Bank holds the Title Deed cards and houses and hotels prior to purchase and use by the players. The Bank pays salaries and bonuses. It sells and auctions properties and hands out the proper Title Deed cards; it sells houses and hotels to the players and loans money when required on mortgages. The Bank collects all taxes, fines, loans and interest, and the price of all properties which it sells and auctions. The Bank never "goes broke." If the Bank runs out of money, the Banker may issue as much more as may be needed by writing on any ordinary paper.

The Play

Starting with the Banker, each player in turn throws the dice. The player with the highest total starts the play: Place your token on the corner marked "GO," throw the dice and move your token in the direction of the arrow the number of spaces indicated by the dice. After you have completed your play, the turn passes to the left. The tokens remain on the spaces occupied and proceed from that point on the player's next turn. Two or more tokens may rest on the same space at the same time. According to the space your token reaches, you may be entitled to buy real estate or other properties--or obliged to pay rent, pay taxes, draw a Chance or Community Chest card, "Go to Jail" etc. If you throw doubles, you move your token as usual, the sum of the two dice, and are subject to any privileges or penalties pertaining to the space on which you land. Retaining the dice, throw again and move your token as before. If you throw doubles three times in succession, move your token immediately to the space marked "In Jail" (see JAIL).

"Go"

Each time a player's token lands on or passes over GO, whether by throwing the dice or drawing a card, the Banker pays him/her a $200 salary. The $200 is paid only once each time around the board. However, if a player passing GO on the throw of the dice lands 2 spaces beyond it on Community Chest, or 7 spaces beyond it on Chance, and draws the "Advance to GO" card, he/she collects $200 for passing GO the first time and another $200 for reaching it the second time by instructions on the card.

Buying Property

Whenever you land on an unowned property you may buy that property from the Bank at its printed price. You receive the Title Deed card showing ownership; place it face up in front of you. If you do not wish to buy the property, the Banker sells it at auction to the highest bidder. The buyer pays the Bank the amount of the bid in cash and receives the Title Deed card for that property. Any player, including the one who declined the option to buy it at the printed price, may bid. Bidding may start at any price.

Paying Rent

When you land on property owned by another player, the owner collects rent from you in accordance with the list printed on its Title Deed card. If the property is mortgaged, no rent can be collected. When a property is mortgaged, its Title Deed card is placed face down in front of the owner. It is an advantage to hold all the Title Deed cards in a color-group (e.g., Boardwalk and Park Place; or Connecticut, Vermont and Oriental Avenues) because the owner may then charge double rent for unimproved properties in that color-group. This rule applies to unmortgaged properties even if another property in that color-group is mortgaged. It is even more advantageous to have houses or hotels on properties because rents are much higher than for unimproved properties. The owner may not collect the rent if he/she fails to ask for it before the second player following throws the dice.

"Chance" and "Community Chest"

When you land on either of these spaces, take the top card from the deck indicated, follow the instructions and return the card face down to the bottom of the deck. The "Get Out of Jail Free" card is held until used and then returned to the bottom of the deck. If the player who draws it does not wish to use it, he/she may sell it, at any time, to another player at a price agreeable to both.

"Income Tax"

If you land here you have two options: You may estimate your tax at $200 and pay the Bank, or you may pay 10% of your total worth to the Bank. Your total worth is all your cash on hand, printed prices of mortgaged and unmortgaged properties and cost price of all buildings you own. You must decide which option you will take before you add up your total worth.

"Jail"

You land in Jail when... (1) your token lands on the space marked "Go to Jail"; (2) you draw a card marked "Go to Jail"; or (3) you throw doubles three times in succession. When you are sent to Jail you cannot collect your $200 salary in that move since, regardless of where your token is on the board, you must move directly into Jail. Your turn ends when you are sent to Jail. If you are not "sent" to Jail but in the ordinary course of play land on that space, you are "Just Visiting," you incur no penalty, and you move ahead in the usual manner on your next turn. You get out of Jail by... (1) throwing doubles on any of your next three turns; if you succeed in doing this you immediately move forward the number of spaces shown by your doubles throw; even though you had thrown doubles, you do not take another turn; (2) using the "Get Out of Jail Free" card if you have it; (3) purchasing the "Get Out of Jail Free" card from another player and playing it; (4) paying a fine of $50 before you roll the dice on either of your next two turns. If you do not throw doubles by your third turn, you must pay the $50 fine. You then get out of Jail and immediately move forward the number of spaces shown by your throw. Even though you are in Jail, you may buy and sell property, buy and sell houses and hotels and collect rents.

<u>"Free Parking"</u>

A player landing on this place does not receive any money, property or reward of any kind. This is just a "free" resting place.

<u>Houses</u>

When you own all the properties in a color-group you may buy houses from the Bank and erect them on those properties. If you buy one house, you may put it on any one of those properties. The next house you buy must be erected on one of the unimproved properties of this or any other complete color-group you may own. The price you must pay the Bank for each house is shown on your Title Deed card for the property on which you erect the house. The owner still collects double rent from an opponent who lands on the unimproved properties of his/her complete color-group. Following the above rules, you may buy and erect at any time as many houses as your judgment and financial standing will allow. But you must build evenly, i.e., you cannot erect more than one house on any one property of any color-group until you have built one house on every property of that group. You may then begin on the second row of houses, and so on, up to a limit of four houses to a property. For example, you cannot build three houses on one property if you have only one house on another property of that group. As you build evenly, you must also break down evenly if you sell houses back to the Bank (see SELLING PROPERTY).

Hotels

When a player has four houses on each property of a complete color-group, he/she may buy a hotel from the Bank and erect it on any property of the color-group. He/she returns the four houses from that property to the Bank and pays the price for the hotel as shown on the Title Deed card. Only one hotel may be erected on any one property.

Building Shortages

When the Bank has no houses to sell, players wishing to build must wait for some player to return or sell his/her houses to the Bank before building. If there are a limited number of houses and hotels available and two or more players wish to buy more than the Bank has, the houses or hotels must be sold at auction to the highest bidder.

Selling Property

Unimproved properties, railroads and utilities (but not buildings) may be sold to any player as a private transaction for any amount the owner can get; however, no property can be sold to another player if buildings are standing on any properties of that color-group. Any buildings so located must be sold back to the Bank before the owner can sell any property of that color-group. Houses and hotels may be sold back to the Bank at any time for one-half the price paid for them. All houses on one color-group may be sold at once, or they may be sold one house at a time (one hotel equals five houses), evenly, in reverse of the manner in which they were erected.

Mortgages

Unimproved properties can be mortgaged through the Bank at any time. Before an improved property can be mortgaged, all the buildings on all the properties of its color-group must be sold back to the Bank at half price. The mortgage value is printed on each Title Deed card. No rent can be collected on mortgaged properties or utilities, but rent can be collected on unmortgaged properties in the same group. In order to lift the mortgage, the owner must pay the Bank the amount of mortgage plus 10% interest. When all the properties of a color-group are no longer mortgaged, the owner may begin to buy back houses at full price. The player who mortgages property retains possession of it and no other player may secure it by lifting the mortgage from the Bank. However, the owner may sell this mortgaged property to another player at any agreed price. If you are the new owner, you may lift the mortgage at once if you wish by paying off the mortgage plus 10% interest to the Bank. If the mortgage if not lifted at once, you must pay the Bank 10% interest when you buy the property and if you lift the mortgage later you must pay the Bank an additional 10% interest as well as the amount of the mortgage.

Bankruptcy

You are declared bankrupt if you owe more than you can pay either to another player or to the Bank. If your debt is to another player, you must turn over to that player all that you have of value and retire from the game. In making this settlement, if you own houses or hotels, you must return these to the Bank in exchange for money to the extent of one-half the amount paid for them; this cash is given to the creditor. If you have mortgaged property you also turn this property over to your creditor but the new owner must at once pay the Bank the amount of interest on the loan, which is 10% of the value of the property. The new owner who does this may then, at his/her option, pay the principal or hold the property until some later turn, then lift the mortgage. If he/she holds property in this way until a later turn, he/she must pay the interest again upon lifting the mortgage. Should you owe the Bank, instead of another player, more than you can pay (because of taxes or penalties) even by selling off buildings and mortgaging property, you must turn over all assets to the Bank. In this case, the Bank immediately sells by auction all property so taken, except buildings. A bankrupt player must immediately retire from the game. The last player left in the game wins.

Miscellaneous

Money can be loaned to a player only by the Bank and then only by mortgaging property.

No player may borrow from or lend money to another player.

Appendix B

An article from http://accounting.smartpros.com/x6862.xml

Teach Accounting With a Monopoly Board

Board Walk and Park Place Were Never Like This!

By: W. Albrecht

October 4, 1999 — Everyone knows how to play Monopoly, but has anyone ever considered its advantages as a simulation game in the classroom to teach financial accounting? Motivating today's student to learn our profession often involves more than book learning. Getting students involved in a game goes a long way to entertaining them -- while intentionally reinforcing academics.

The Need for Different Methods

In 1993, I became increasingly aware that many accounting educators, practitioners and students agreed that accounting education was excessively procedure and knowledge-oriented. New instructional approaches were called upon to actively engage students, not only to learn accounting, but to provide the know-how to transfer their learning to the real world's complex financial decisions.

I also was searching for instructional methods that would improve my own teaching effectiveness. Not much learning takes place in the traditional lecture method (so frequently used in accounting courses). Discussion-based classes do better, but the best learning takes place when students must make their own decisions after analyzing and evaluating unique situations.

It became apparent that using a simulation game could be one part of a solution. A simulation game imitates some part of reality, yet is a contest. Simulation games offer the following advantages:

- Motivate many students to participate to a greater degree than they would in a lecture or discussion class.
- Increase the ability to recall factual knowledge and improve problem-solving skills.
- Offer students participating in simulation games experience greater attitudinal change than those engaging in more traditional learning environments.
- Offer the potential for students to attribute greater value to accounting information in the decision-making process.
- Give students, through the "wheeling and dealing" simulations, intensive practice in verbal and written communication.
- Require flexibility in thinking as students adapt to a dynamic environment.
- Benefit students with varying skills and experience because participants can play at their own level.

In fleshing out this idea, I came across an article in Issues in Accounting Education by Robert Kneckel, describing how he had students play Monopoly and then prepare an Income Statement and Balance Sheet at the conclusion of the game. I was intrigued by the thought of using the game, but felt that two modifications might improve the instructional effectiveness of its use.

First, I reasoned that it would be better for students to prepare financial statements at regular intervals during the game -- feedback from earlier assignments could be incorporated by a student into new assignments. Also, working in the accounting environment during the game would give students an opportunity to apply accrual accounting. And, since the purpose of the game is to drive everyone else into bankruptcy, the financial statements would pretty much look the same. In addition, Kneckel's approach gave students no opportunity to use the accounting information to make decisions.

Playing the Game
I incorporated Monopoly basics to create a simulation game called "Real Money." In Real Money, students are assigned to games so that there are four tokens per game. Students play 13 turns (one year or accounting period) of Monopoly, writing down everything that happens to them during the game.

At the conclusion of the first year, the students each prepare a set of financial statements for their token, with adjusting entries for income tax (a percentage of net income), interest, depreciation on houses and hotels, and accrued salary for progress around the board. The financial statements are submitted for grading, and I assume the role of public auditor.

After grading the first year financial statements, I transcribe them into a spreadsheet and upload it to the course Web page, http://www.profalbrecht.com/classes/acct321/realmoney/. Students then must view and analyze the four sets of financial statements for each game. Each student is allocated an imaginary $100 per game for investment. Essentially, students must wager on whom they think will eventually win each game.

In a typical game, about two-thirds of the properties have been acquired by the end of the first year, but there are no monopolies. The investment patterns show wide-spread diversification. If a token is perceived to have a slight advantage based on total property acquired, then an average investment or wager will be $30-35. Those tokens perceived to be behind will have averaged $15-20 of investment support.

After I collect each student's investments or bets, the class is instructed to play a second year of Monopoly and submit financial statements. After grading the financial statement for all tokens in each game, I publish them again on the course Web page.

Again, students receive $100 of investment per game. In a typical game, most of the properties have been acquired, and some monopolies have been created as a result of trades. Average investments or bets range from about $50 for tokens with the greatest perceived advantage in properties and houses, to $10 for tokens with the least future earnings capabilities.

Next, students play a third year of Monopoly, submit financial statements and engage in another round of betting with an additional $100 per game. There are two patterns of game results reflected in the financials. In some games, there are two or three competing tokens with monopolies, and investments in houses and hotels. Financial distress in the losing companies is very evident. In other games, all tokens are still very close. In the games with two or three leading companies' tokens, amounts run an average of $40 to $90. In the games with equal earning tokens, amounts average about $25.

The fourth and final year of Monopoly is played, and students submit financial statements for grading. A winner for each game is declared based on a highest retained earnings. The wagers for each year are tallied and winnings are presented to each student that invested on the winning token.

What They Learn
Students learn to evaluate income statements for the quality of reported earnings. They base their estimates of investment value on the chance of future success instead of documented past success. Also, if certain teams are slow to capitalize on early advantages, investors turn from them in later bidding rounds. The accounting portion contains all the advantages listed by Knechel. In addition, students get to repeat the end of period accounting cycle, and hopefully, correct any errors that may have occurred during their first attempt.

This simulation class can be used in financial accounting courses at many levels. It could effectively be used as a review in Intermediate Accounting or to supplement an introductory course. It can also be used in MBA classes. I generally do not force MBA students to prepare journal entries, allowing them the option of analyzing the effects of transactions through use of a detailed balance sheet equation. Financial statements are then prepared from this analysis.

Appendix C: Network Monopoly – User Guide

Introduction

Network Monopoly is a network-based application that allows players in the different computers in a network to play the game. It can then applied to be the web-based application that manages more wide range of players.

Objectives

These are the main objectives of Network Monopoly:

- Provides a great game play to the users.
- Provides a way to play with friends and family.
- Provides social interaction among players through message exchange features in the system.
- Provides a great framework for human interaction with all the wheeling and dealing.
- Provides a way of education to the children.

About This Manual

This user manual gives the users step-by-step guide through the application, from setting up to using core function.

- Getting Started (Deployment Guide)
- The Server Environment

- The Client Environment
- Server Guide
- Client Guide

Getting Started (Deployment Guide)

In this section, the following will be discussed: System Requirement, Server Deployment, and Client Deployment.

System Requirement

For Network Monopoly function properly, the following requirements are recommended for the system:

Hardware:

Intel Pentium Processor or Compatible and above

32 MB RAM or above

Network Interface Card that connect to Network

Hard Disk Space of At least 20MB of free Space

SVGA Graphics Adapter

Keyboard & Mouse

Software:

Window 95/98, Window NT 4.0 Server or Window 2000

Java Runtime Environment 1.2 or above (available for free download at http://java.sun.com)

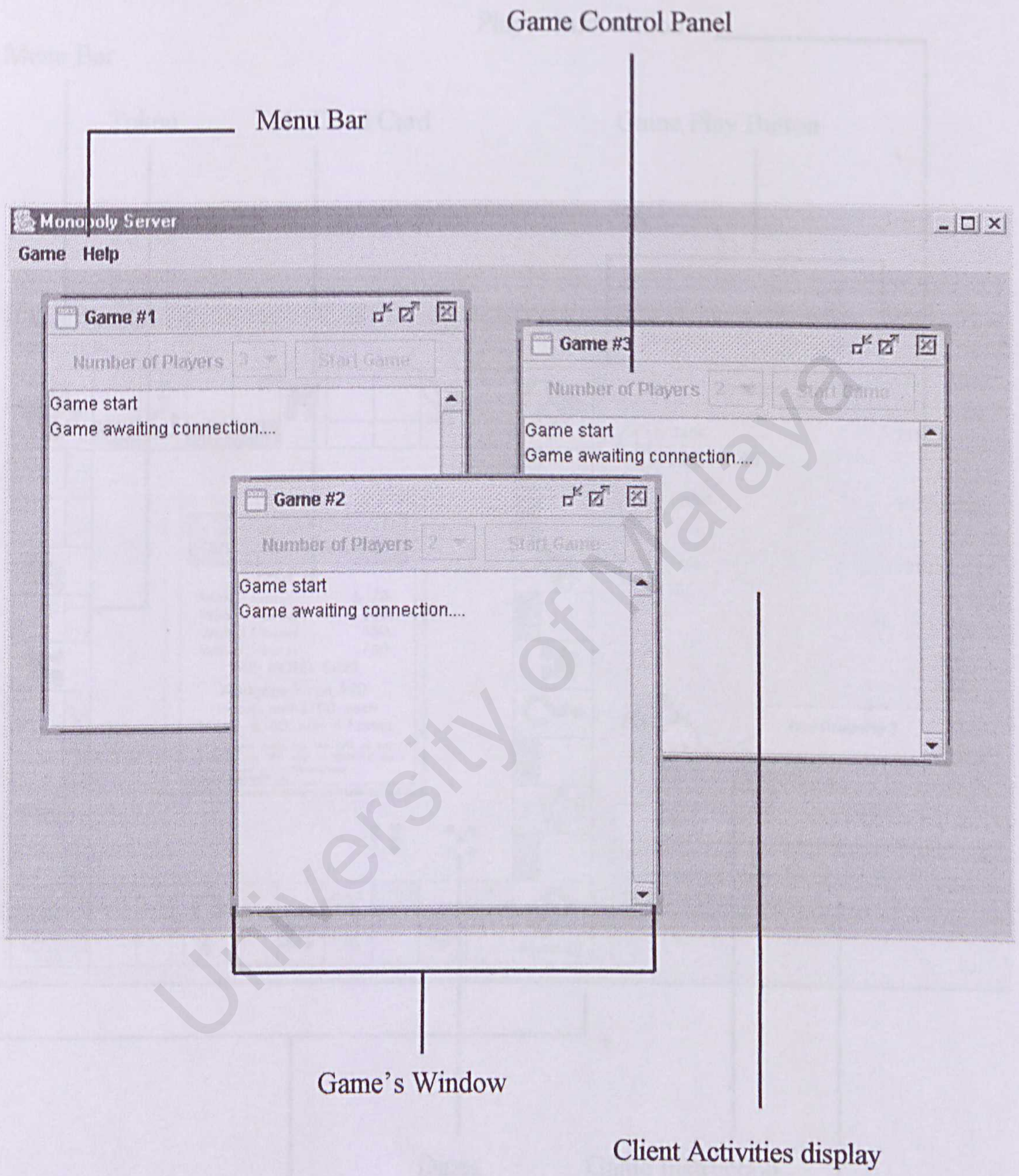
Server Deployment

1. Select a computer as server computer. For network monopoly to function properly, it needs a center control computer to maintain the connection between clients. So, the selected center computer does not really need a server, a personal computer also can provides the same performance.
2. Insert Floppy Disk 1 into Drive A of the selected computer.
3. Copy the Server Folder into you desired drive.

Client Deployment

1. Select a computer within the network as client computer.
2. The selected computer must connected to the server computer.
3. Insert Floppy Disk 1 into Drive A
4. Copy the Monopoly Folder in the desired drive.

The Server Environment

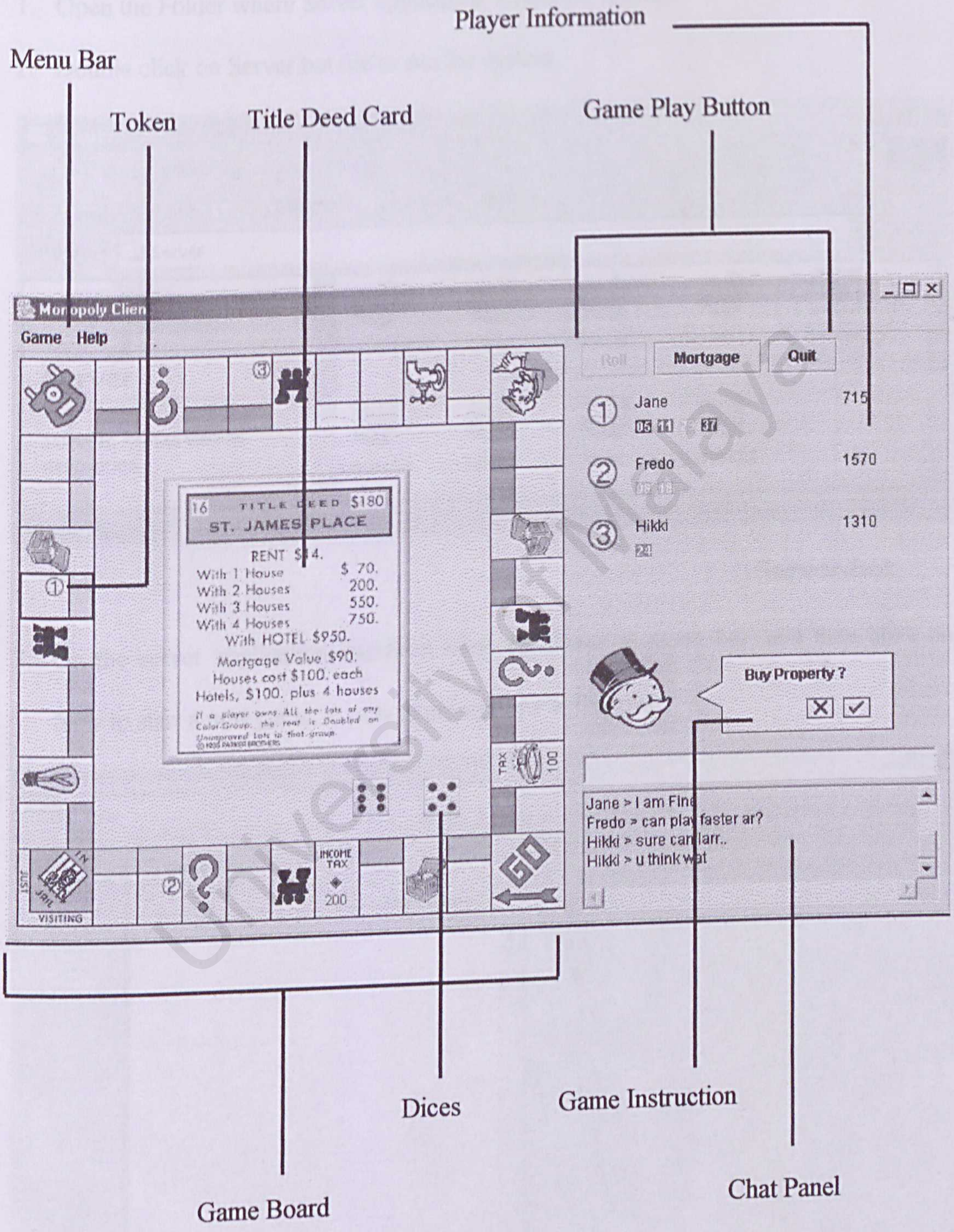


Game start
Game awaiting connection....

Game start
Game awaiting connection....

Game start
Game awaiting connection....

The Client Environment



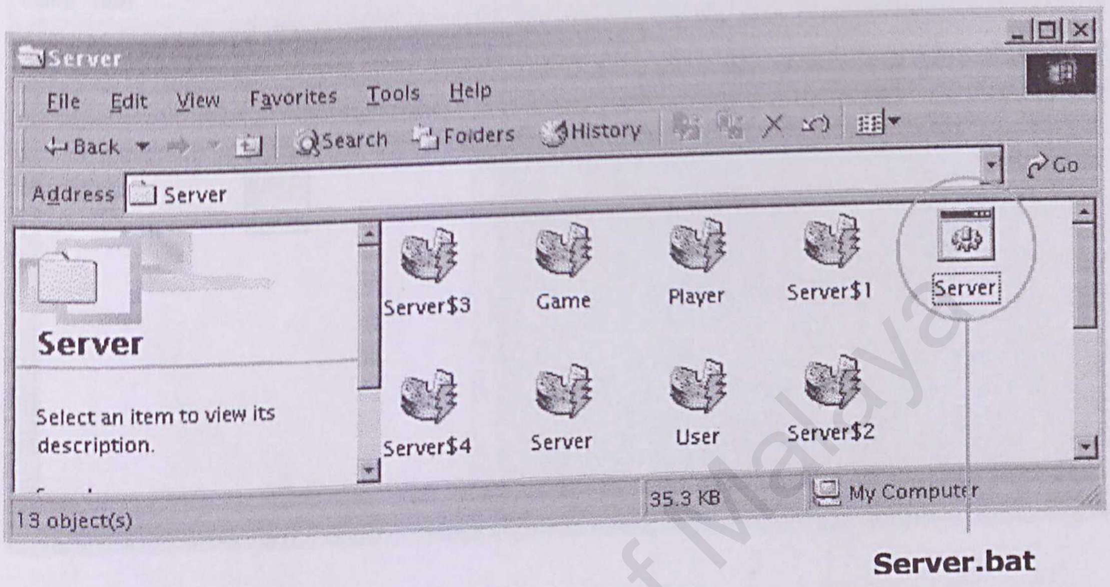
16 TITLE DEED $180
ST. JAMES PLACE
RENT $14.
With 1 House $ 70.
With 2 Houses 200.
With 3 Houses 550.
With 4 Houses 750.
With HOTEL $950.
Mortgage Value $90.
Houses cost $100. each
Hotels, $100. plus 4 houses

①	Jane	715
②	Fredo	1570
③	Hikki	1310

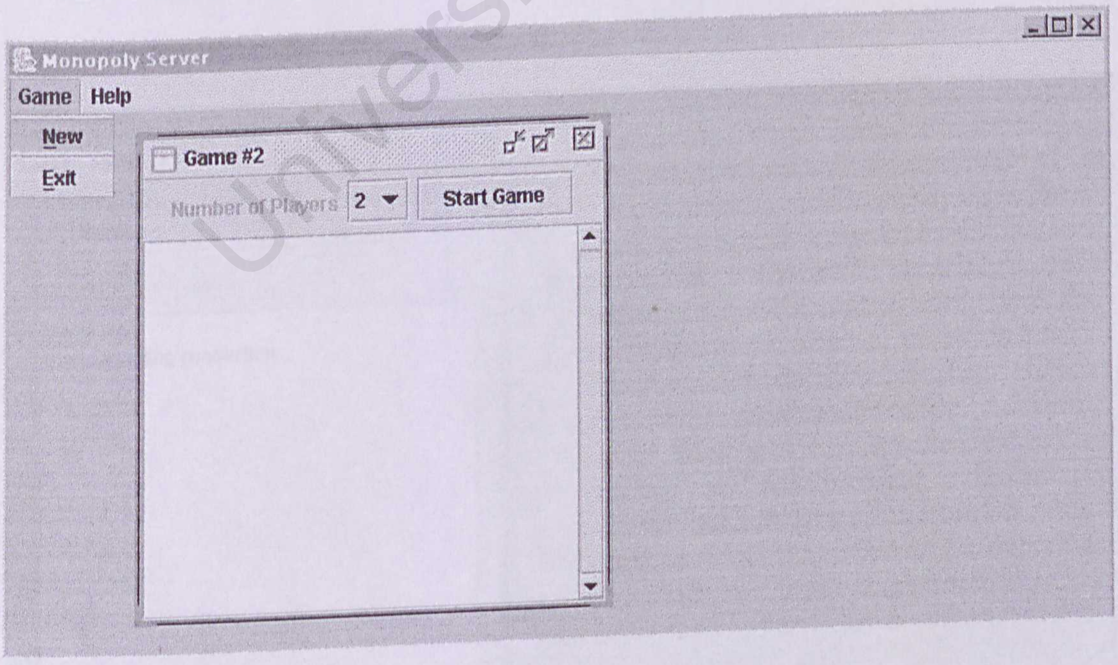
Jane > I am Fine
Fredo > can play faster ar?
Hikki > sure can larr..
Hikki > u think wat

Server Guide

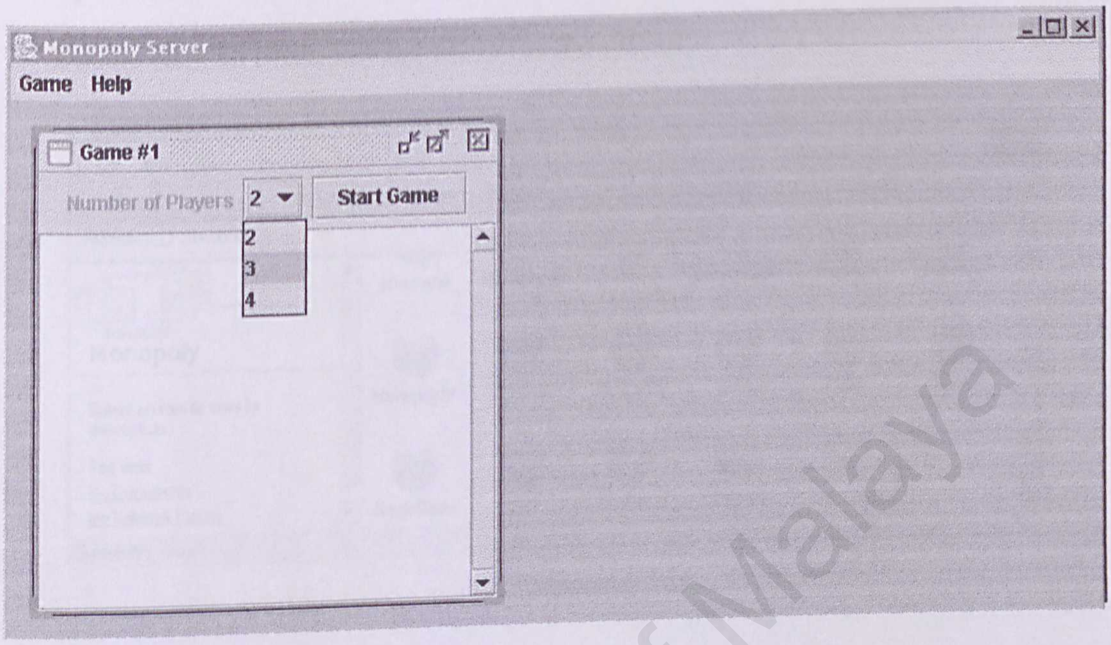
1. Open the Folder where Server application's files are resides.
2. Double click on Server.bat file to run the system.



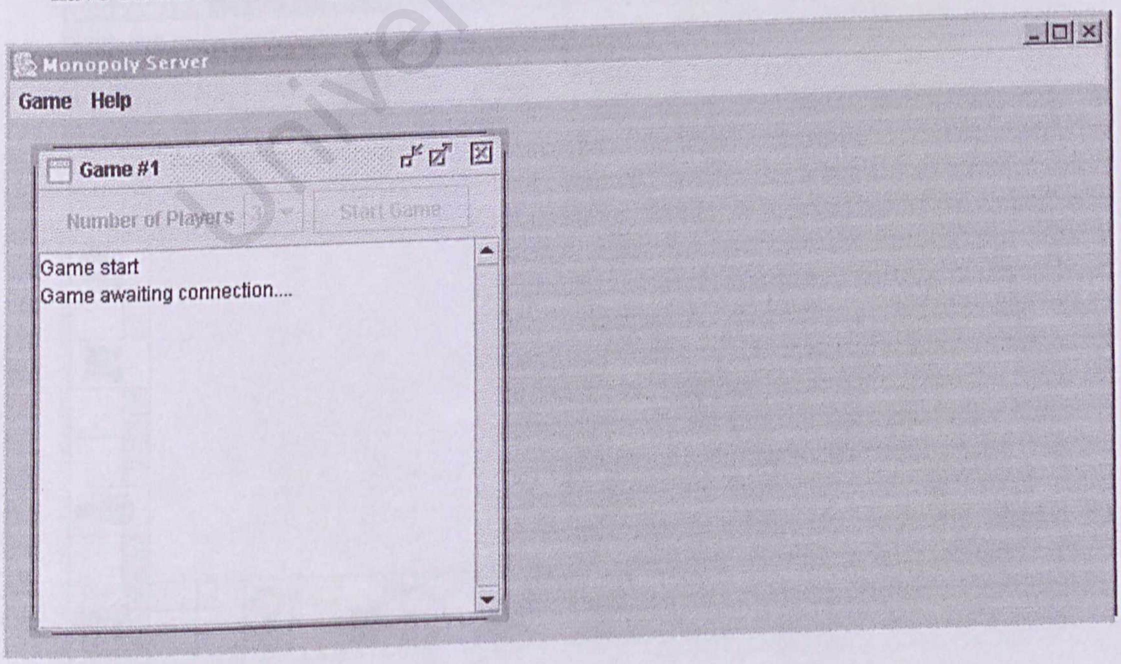
3. On the server application interface, click on **Game** at menu bar, and then click on **New** to start a new game. A new game window will appear.



4. On the Game Window interface, select the number of players to play the game and then click on **Start Game** button to start the game.



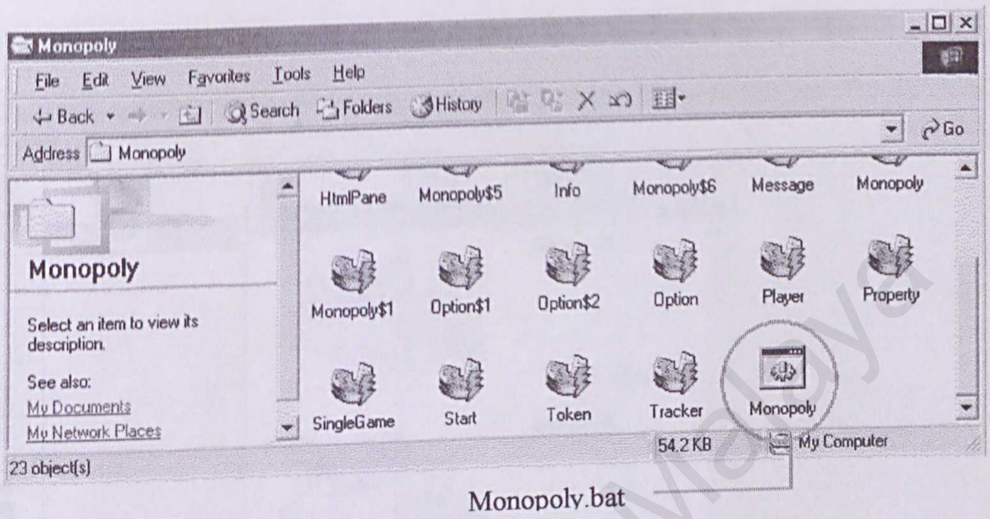
5. After the game started, messages appear to state that the game is start and is now waiting for connection from the client. The game will not started until all the players have connected to the game.



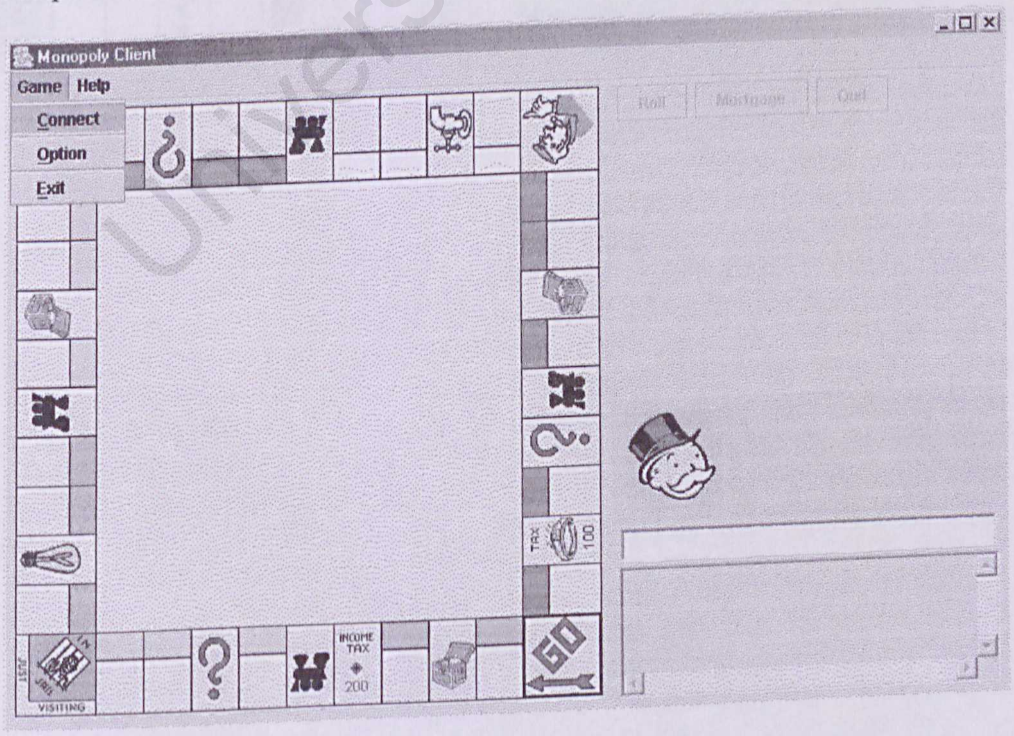
```
Game start
Game awaiting connection....
```

Client Guide

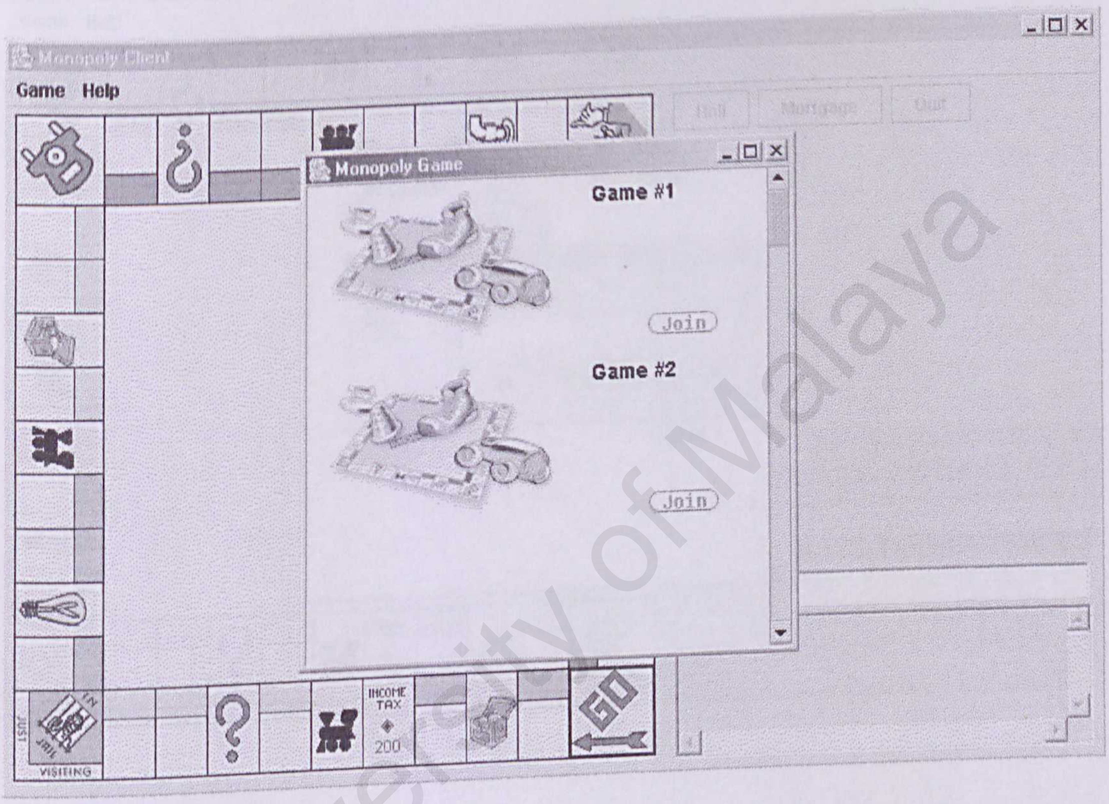
1. Open the Folder where Monopoly Client application's files are resides.
2. Double click on Monopoly.bat file to run the system.



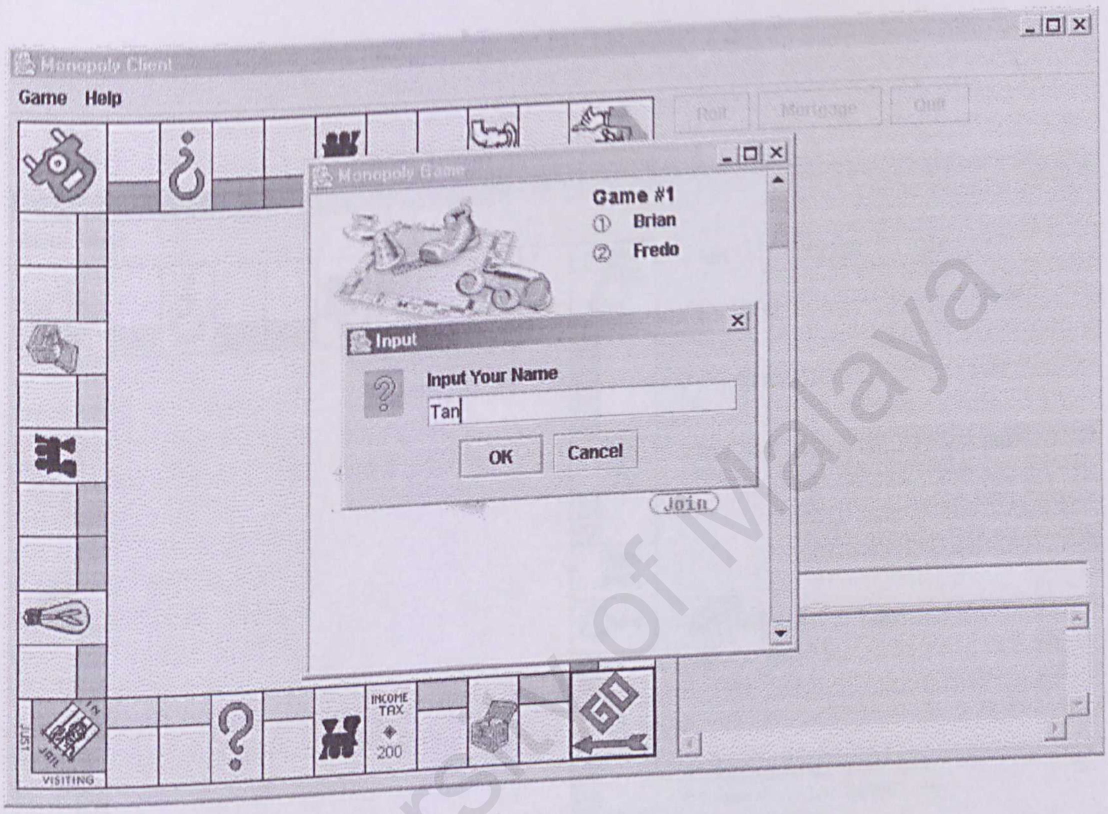
3. On the Monopoly application interface, click on **Game** at menu bar, and then click on **Connect** to join the game. This will connect the client computer to the server computer.



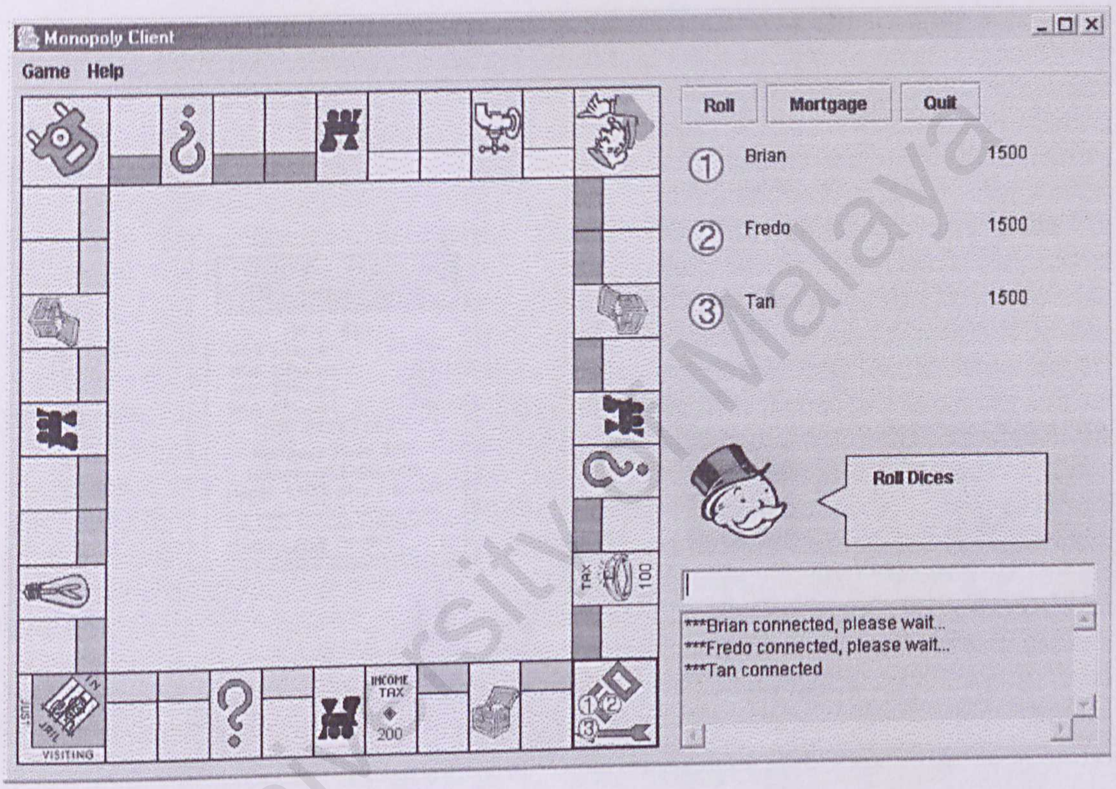
4. A small window will appear. The window displays all the games that are currently running. If the game is available for joining, a join image button will show. Choose a game that available and then click on the button to join.



5. After that, a window appears for you to key in your name in the game. Key in the name and then click on **OK** button to proceed to the game.



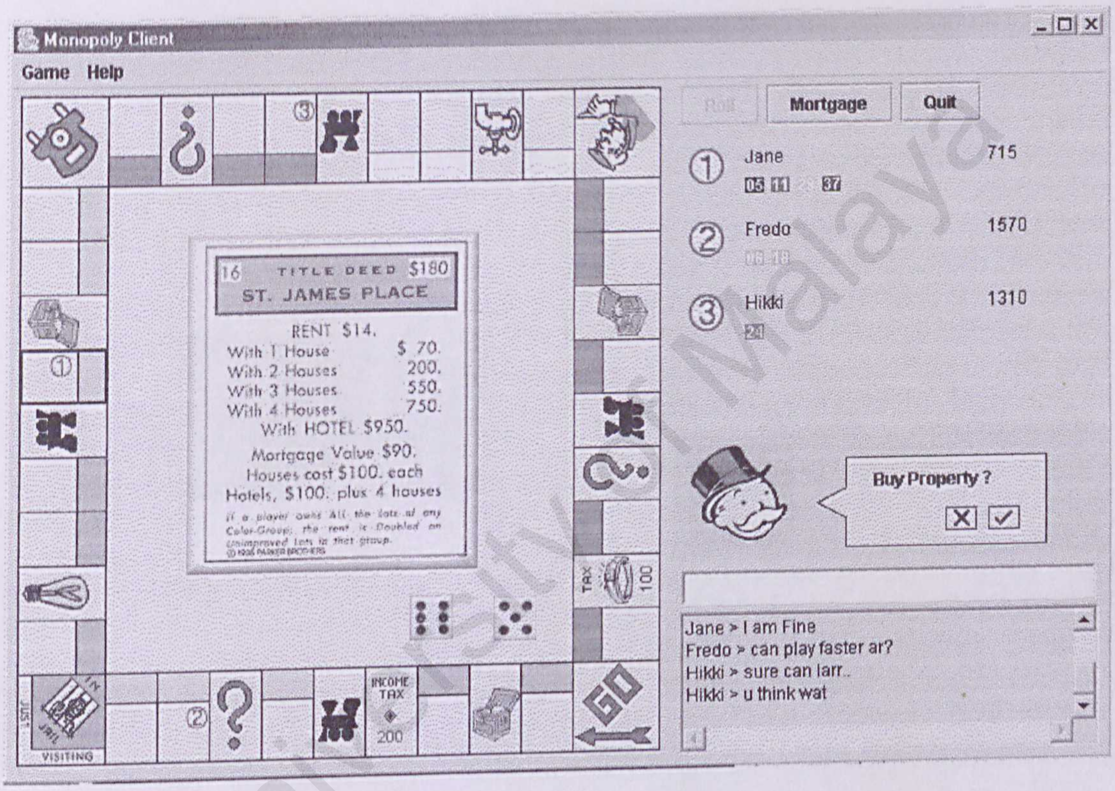
6. The game will not start until all the players are connected to the game. After all players are connected, the first player will start the game. Game Instruction "**Roll Dices**" will show indicates that player can start to roll the dices. Click on **Roll** button to roll dices.



①	Brian	1500
②	Fredo	1500
③	Tan	1500

```
***Brian connected, please wait...
***Fredo connected, please wait...
***Tan connected
```

7. The token will moves according to the number show on the dices. If token landed on an unowned property, a **Title Deed Card** for that property appears on the center of the game board. Game Instruction "**Buy Property**" show to indicate that the property is unowned and you may buy it by click on the "√" button.

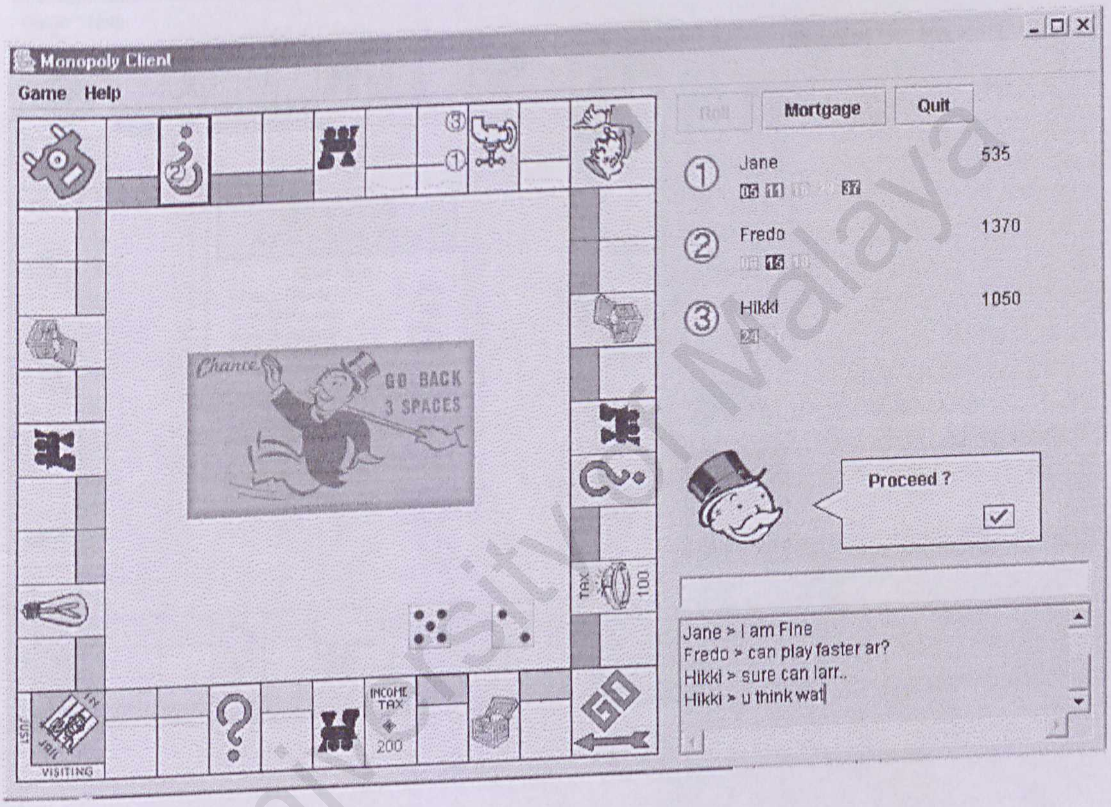


16 TITLE DEED $180
ST. JAMES PLACE
RENT $14.
With 1 House $ 70.
With 2 Houses 200.
With 3 Houses 550.
With 4 Houses 750.
With HOTEL $950.
Mortgage Value $90.
Houses cost $100. each
Hotels, $100. plus 4 houses
If a player owns ALL the lots of any Color-Group, the rent is Doubled on Unimproved lots in that group.
© 1935 PARKER BROTHERS

①	Jane	715
②	Fredo	1570
③	Hikki	1310

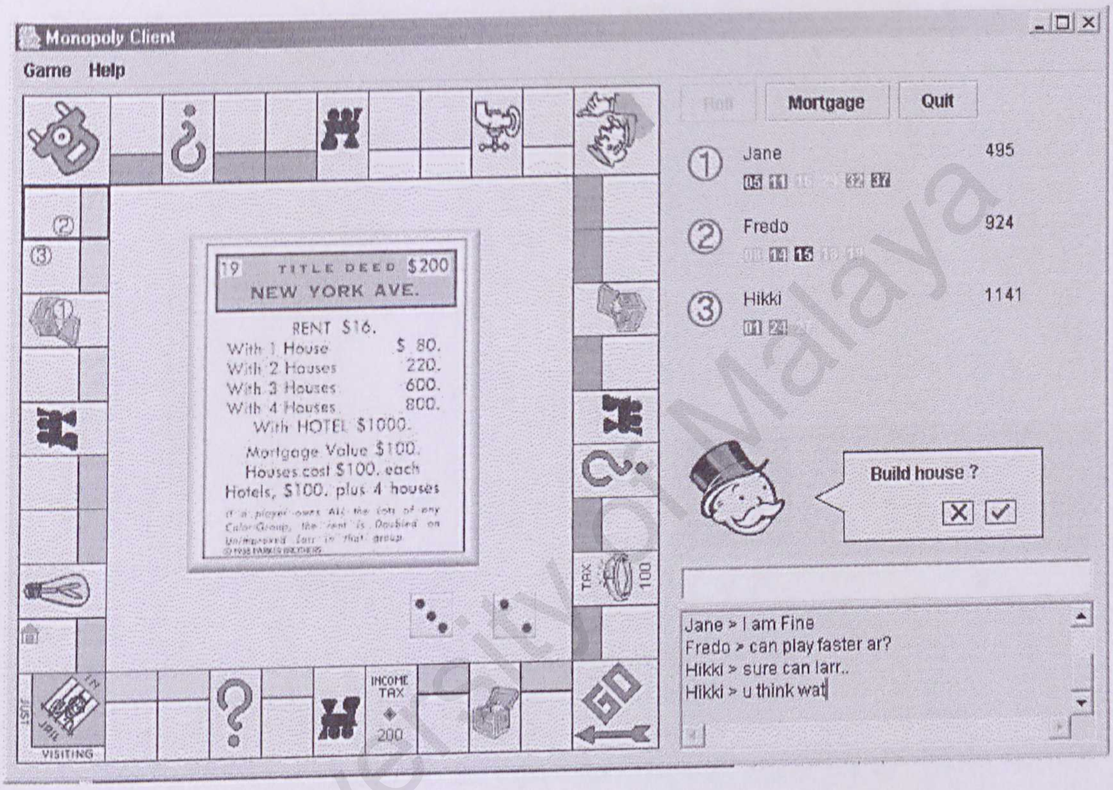
Jane > I am Fine
Fredo > can play faster ar?
Hikki > sure can larr..
Hikki > u think wat

8. If the token landed on the **Chance** or **Community Chest**, a card will show on the center of the board. The card is randomly picked from all 16 cards. Game Instruction "**Proceed**" show. Click on "√" to proceed to the action show on the card.



Jane > I am Fine
Fredo > can play faster ar?
Hikki > sure can larr..
Hikki > u think wat

9. If a player landed on the property that owned by itself, he may build house on that property. Game Instruction "**Build house**" show. Click on "√" to build the house. If the property already has 4 houses, the next house build will be a hotel.



19 TITLE DEED $200
NEW YORK AVE.

RENT $16.

With 1 House	$ 80.
With 2 Houses	220.
With 3 Houses	600.
With 4 Houses	800.

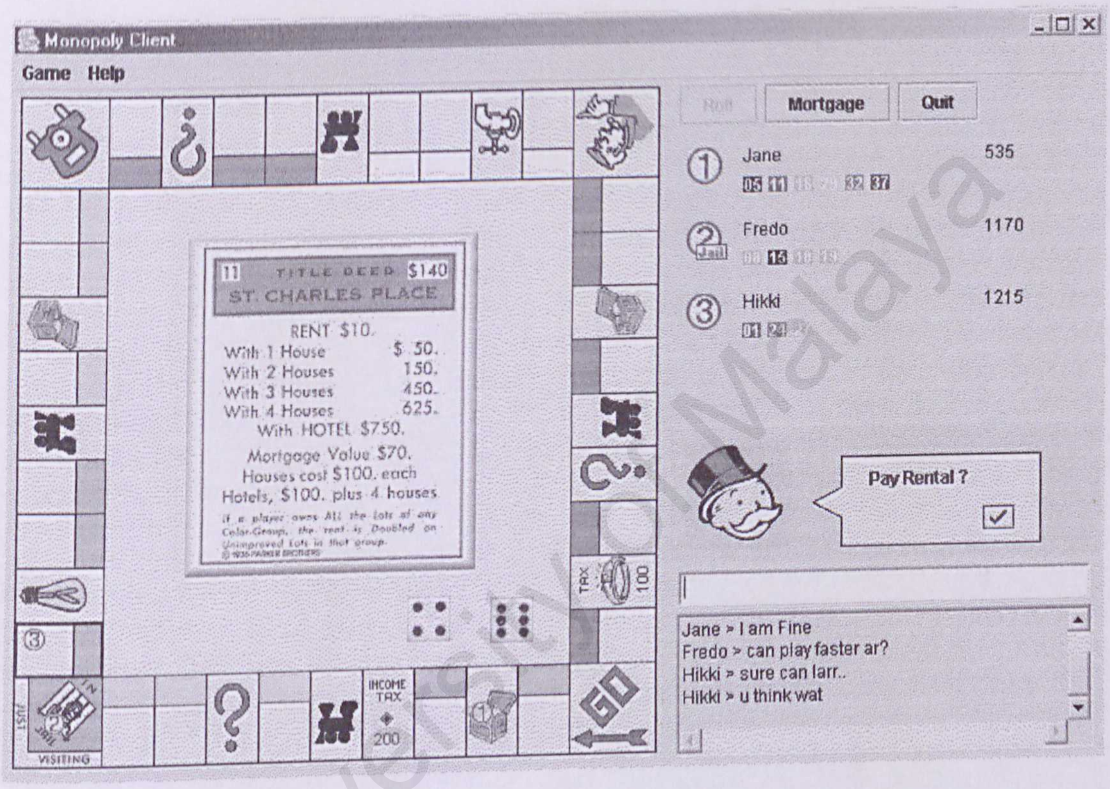
With HOTEL $1000.

Mortgage Value $100.
Houses cost $100. each
Hotels, $100. plus 4 houses

If a player owns ALL the lots of any Color-Group, the rent is Doubled on Unimproved lots in that group.
©1935 PARKER BROTHERS

```
Jane > I am Fine
Fredo > can play faster ar?
Hikki > sure can larr..
Hikki > u think wat
```

10. If player landed on property that owned by other players, he/she need to pay rental according to the rental show in the **Title Deed Card**. Game Instruction "**Pay Rental**" show. Click on "√" to pay the rental.

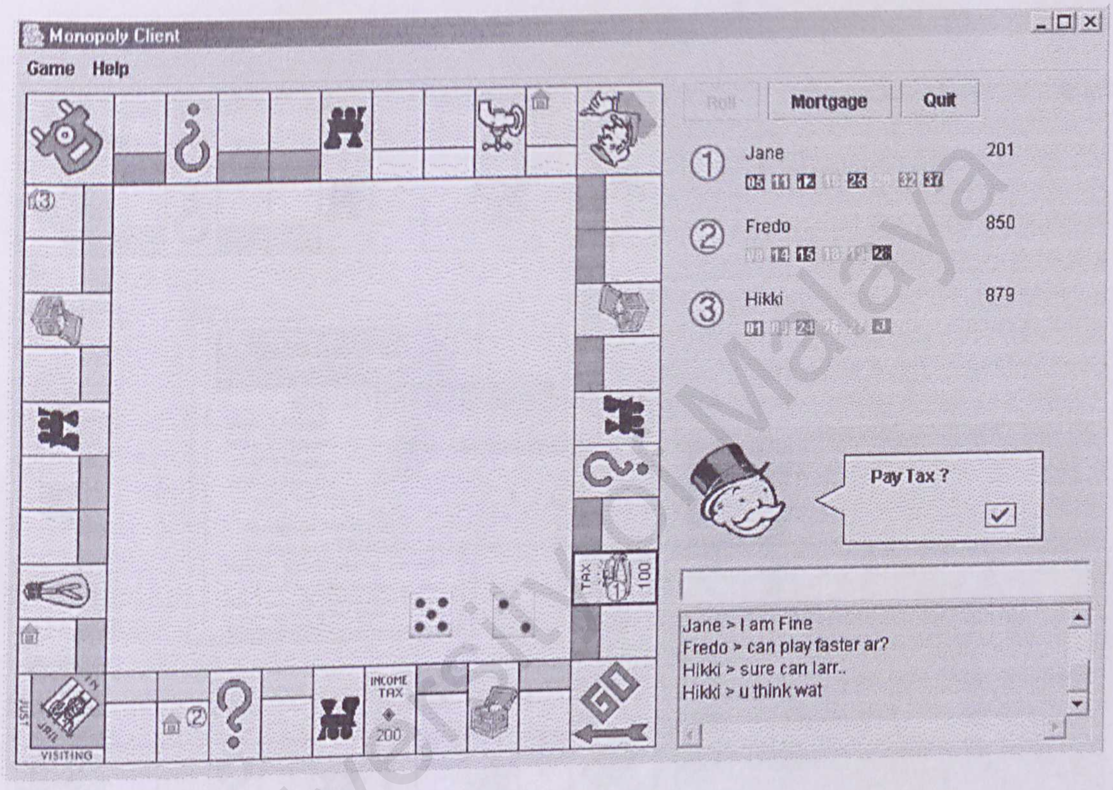


11 TITLE DEED $140
ST. CHARLES PLACE
RENT $10.
With 1 House $ 50.
With 2 Houses 150.
With 3 Houses 450.
With 4 Houses 625.
With HOTEL $750.
Mortgage Value $70.
Houses cost $100. each
Hotels, $100. plus 4 houses
If a player owns All the lots of any Color-Group, the rent is Doubled on Unimproved Lots in that group.

①	Jane	535
②	Fredo	1170
③	Hikki	1215

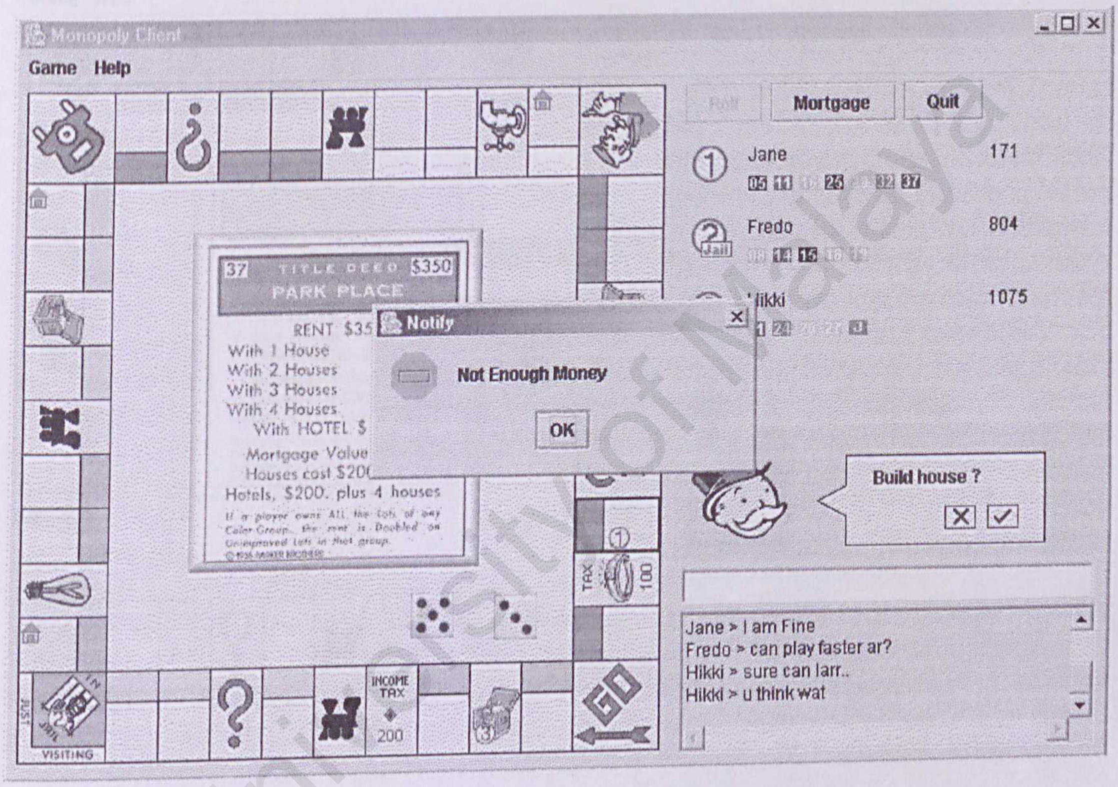
Jane > I am Fine
Fredo > can play faster ar?
Hikki > sure can larr..
Hikki > u think wat

11. If a player landed on the **Tax** or **Income Tax**, he/she needs to pay the tax according to the amount of tax show. Game Instruction "**Pay Tax**" show, click on "√" to pay the tax.



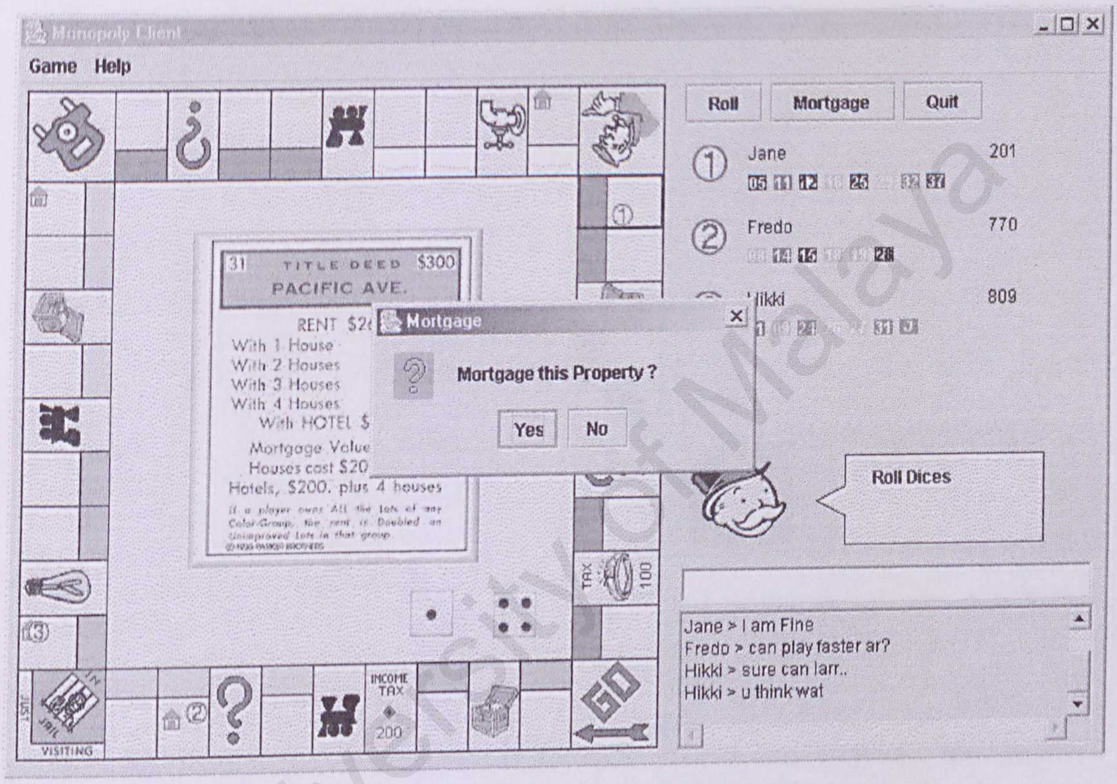
Jane > I am Fine
Fredo > can play faster ar?
Hikki > sure can larr..
Hikki > u think wat

12. If player do not has enough money to pay the payment. **Not Enough Money** window will show. Player can mortgage his/her assets to get the enough money to pay. If player still do not has enough money after mortgage all assets, he/she will declared **bankrupt.**



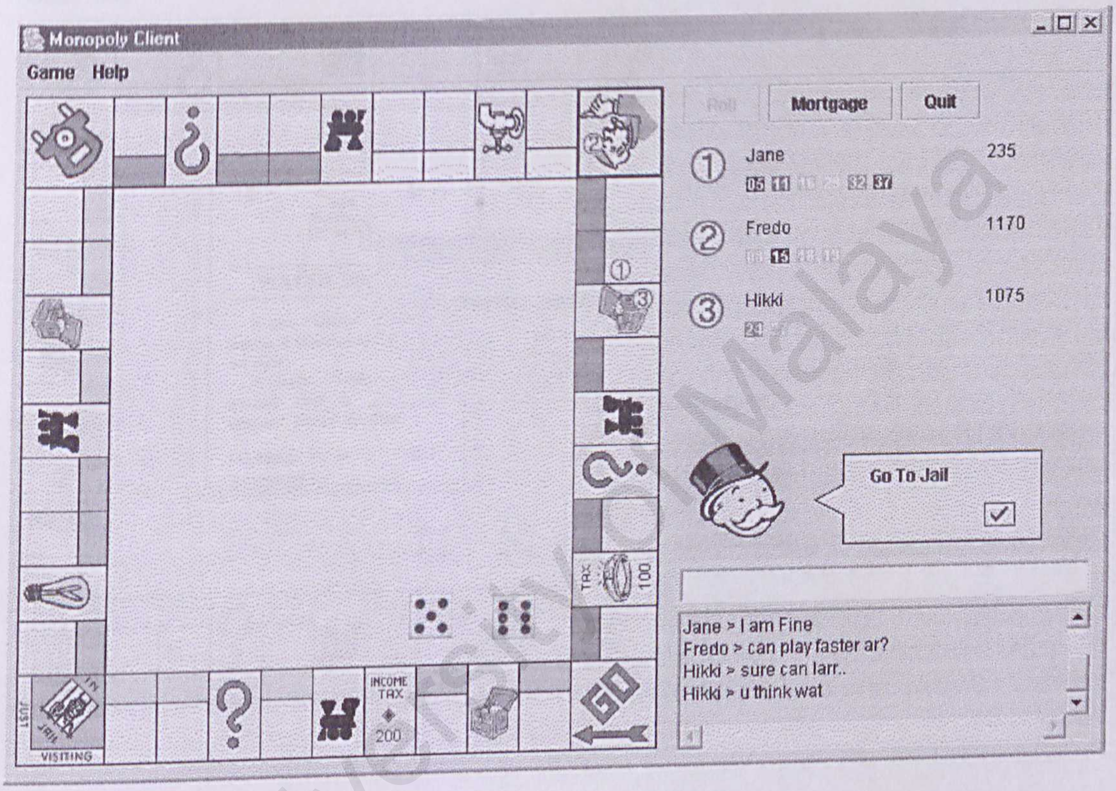
Jane > I am Fine
Fredo > can play faster ar?
Hikki > sure can larr..
Hikki > u think wat

13. To mortgage property, click on the property that you owned on the game board, after that property is selected, click on **Mortgage** button to mortgage that property. A window will show to confirm you mortgage action.



Jane > I am Fine
Fredo > can play faster ar?
Hikki > sure can larr..
Hikki > u think wat

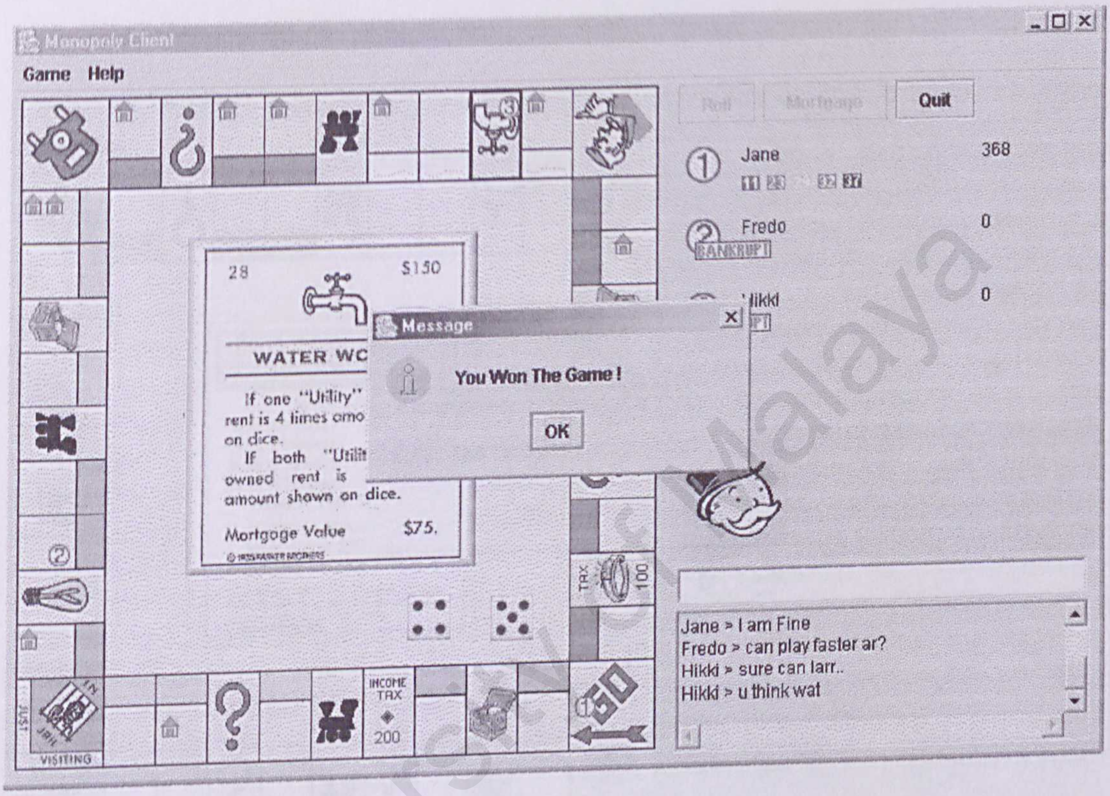
14. If a player landed on the picture of a police on the board, he/she will directly send to the Jail. Game Instruction "**Go to Jail**" show to indicate that you are send to Jail. Click on "√" to go to Jail.



Jane > I am Fine
Fredo > can play faster ar?
Hikki > sure can larr..
Hikki > u think wat

15. If a player plays until all the players of the game bankrupt, he/she will win the game.

A window is show to indicate that player has won the game.



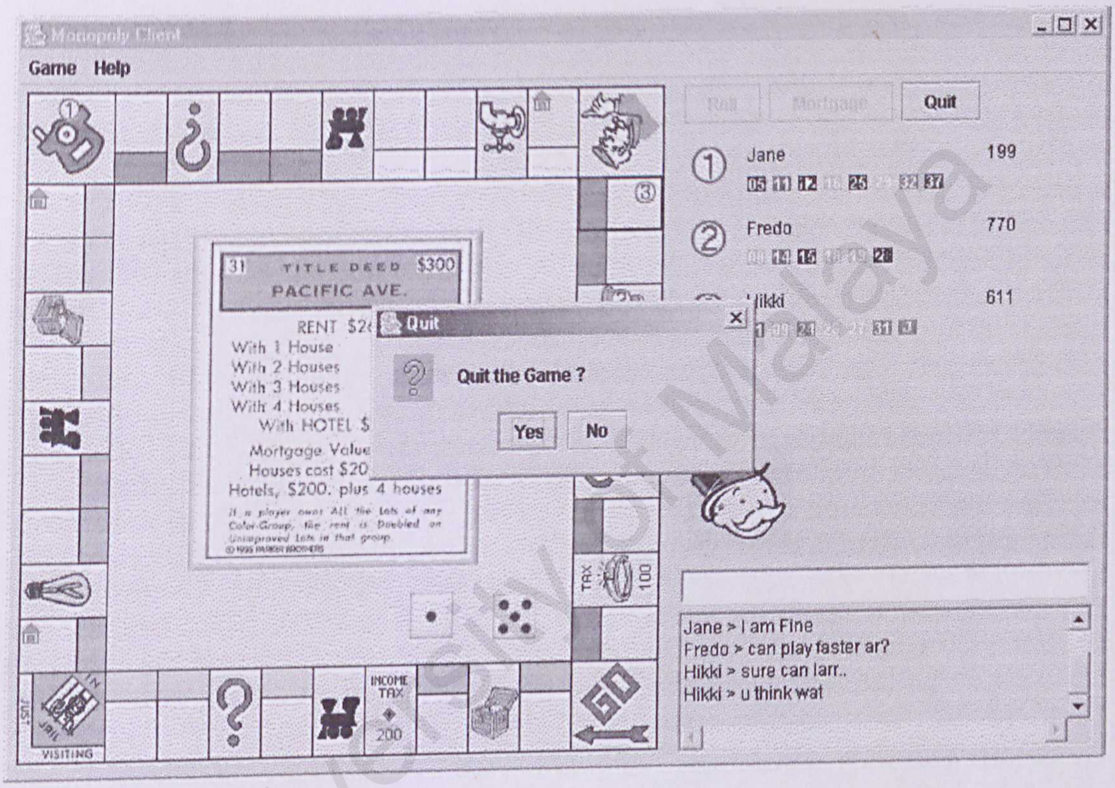
Jane 368

Fredo 0 BANKRUPT

Hikki 0

Jane > I am Fine
Fredo > can play faster ar?
Hikki > sure can larr..
Hikki > u think wat

16. Player can quit the game during the game play. Just click on **Quit** button to quit from the game. A window will show to confirm you quit action. Click on **Ok** to proceed to quit action.



Jane > I am Fine
Fredo > can play faster ar?
Hikki > sure can larr..
Hikki > u think wat