

CENG 491

**Computer Engineering
Design 1**



**Requirements Analysis
Report**

**Geeks in
Action**



Table of Contents

1. INTRODUCTION	5
1.1 System Purpose	5
1.2 System Scope	5
1.3 Goals of the Team	5
1.4 Process Model	6
1.5 Team Organization	6
2. RESEARCH	7
2.1 Market Research:	7
2.2 Literature Survey and Technical Research:	9
2.2.1 Graphics:	9
2.2.2 Sound:	11
2.2.3 Artificial Intelligence:	12
2.2.4 Network:	12
2.3 Game Engine:	13
3 DESCRIPTION	14
3.1 Game Modes	14
3.1.1 Traditional Mode	14
3.1.2 Japanese Goal Mode	15
3.1.3 Street Football Mode	15
3.1.4 For Fun Modes	15
3.1.5 Training Mode	16

Requirements Analysis Report

Geeks in Action

3.1.6 Edit Mode	17
3.2 Weather Condition	19
3.3 Enviroment Properties	19
3.4 Sound Properties	20
4 REQUIREMENTS.....	21
4.1 Functional Requirements	21
4.1.1 Menu Requirements.....	21
4.1.2 In Game Functions.....	23
4.2 Structural Requirements	24
4.2.1 Graphics.....	24
4.2.2 Sound.....	24
4.2.3 Network.....	24
4.2.4 Artificial Intelligence (AI)	25
4.2.5 Physics	25
4.3 Performance Requirements	25
4.4 Software Requirements.....	25
4.5 Hardware Requirements	26
4.6 Non Functional Requirements.....	26
4.6.1 Security	26
4.6.2 Maintainability	26
4.6.3 Portability	26
4.7 Project Constraints	27
4.7.1 Time Constraints.....	27
4.7.2 Language Constraints	27

Requirements Analysis Report

Geeks in Action

4.7.3 Data Constraints	27
4.7.4 User Interface Constraints.....	27
5 SYSTEM ANALYSIS AND MODELING	27
5.1 Functional Modeling.....	27
5.1.1 Level 0 of Data Flow Diagram.....	27
5.1.2 Level 1 of Data Flow Diagram.....	28
5.2 Use Case Analysis	30
5.2.1 Start Menu Use Case	30
5.2.2 In Game Menu Use Case	31
6 Gantt Chart	32

1. INTRODUCTION

The basic step of design is requirement analysis part. In this part, the below objectives has to be described. The report includes detailed definition of problems and further analysis on functional and technical requirements. Report also includes system models and management plans.

1.1 System Purpose

Project purpose is creating a 3D football game for Linux, in which players will spend on enjoyable time while playing our game.

1.2 System Scope

Scope of the project is limited with:

- Providing an enjoyable football game for linux.
- Analysis of user requirements
- Rendering real world objects and designing 3D computer graphics
- Using a game engine
- Implementation of a basic artificial intelligence
- Support multi player mode
- Support online game modes
- Include sound libraries for various sound effects

1.3 Goals of the Team

Our goals are:

- Learning how to manage project.
- Using documentation efficiently.
- Using time efficiently.
- Learning teamwork.
- Produce a satisfactory software.

1.4 Process Model

We are going to use Scrum design model in this project, which is commonly used with agile software development methodology. It involves iterative and incremental development and we are going to make use of it frequently while developing AI and graphics part of the project. Since it defines strict and irreversible steps, Waterfall model cannot be applied to our project.

We are going to perform an object oriented approach in the project development progress. We are going to focus on the modularity to provide efficient development in future. Since this project will be open source and our game is going to be uploaded to Ubuntu repository, we are going to create components and packages in a way to provide Linux developers easy to understand and develop our project.

1.5 Team Organization

Since every member has nearly same experience about design patterns and concepts of this project, decisions about the project are going to be made by group consensus. This situation yields our team to have democratic decentralized. Communications within the group is going to be horizontal. Like many other software engineering projects, presence of a team leader is vital in

game projects. We cannot evade this fact, so we chose a team leader for coordination and interactions within our team. While making decisions, our team leader will consult each team member in order not to go against our team structure.

2. RESEARCH

2.1 Market Research:

Game sector is very huge and needs big investments. Since Linux is an open source operating system and most of the software developed in Linux environment is not commercial, companies in game development sector disregard Linux platform. This attitude leads up to Linux environment lacking a good and fun football game. In this part our aim is to present the overview of some football games produced for Linux.

BattleBall

BattleBall is essentially the game of soccer, played with military vehicles rather than with people. Each player drives a tank or flies a helicopter, and tries to move the ball down the playfield to the other team's goal. Relatively unlimited number of human or computer players can compete in teams or head-to-head.

SlamSoccer

Slam Soccer 2006 is a football game in 3D comic style which is freeware and open source. It also contains these features.

- Keyboard and gamepad control
- 2 player mode
- Career and world cup

- Register in the online hall of fame
- Build your own stadium
- 80 teams
- 20 stadiums
- 10 weather conditions
- 50 adboards
- 10 referees
- 9 commentators (5 German, 2 English, 2 French)
- 3 languages: German, English, French

Bygfoot

Bygfoot allows you to manage a team by training the players, buying and selling them, contracting loans, maintaining the stadium, etc. You can be promoted or relegated, even become a champion. It includes a "team editor" which you can use to create your own teams or edit the existing ones. The game is deliberately kept simple (though not easy): you only have to keep one eye at your players and another at your stadium.

Eat the whistle

Eat The Whistle is an arcade soccer game similar to famous Amiga titles such as Kick Off or Sensible Soccer. It features several game modes where you can play either as the whole team or as a single player, and you can also manage teams that take part in cups and leagues. There is even an arcade mode with powerups and bonuses, like in the game SpeedBall 2. Eat The Whistle features 30 different field types and numerous sound effects. The game is viewed from the side and

can be controlled with either a joystick or the keyboard. Most in-game settings are configurable, such as the pitch, weather and game daytime, which will impact on the gameplay. There is a replay mode that lets you load and save best moments, a game tactics editor, and teams from the game Sensible World of Soccer can be directly imported.

2.2 Literature Survey and Technical Research:

2.2.1 Graphics:

Our game will be a 3D modeling football game and for that reason nearly most important part of is graphics section. Since this project part challenging and time consuming, we made a research on the internet and decided to use a graphic engine for modeling instead of writing code. Here is some engines, we think to use one of them;

- **OGRE:** OGRE (Object-Oriented Graphics Rendering Engine) is a scene-oriented, cross-platform, flexible 3D engine for hardware-accelerated 3D graphics software. It is licensed under LGPL. (<http://www.ogre3d.org>). It only provides graphic capabilities but it doesn't impose restrictions on other aspects of software so any library can easily be used with it. It also has a large and active community and extensive documentation which complements its design driven approach. The engine can take advantage of latest hardware for its advanced features. Simple API eases overall integration. Since we have specific needs which cannot be met with an all-in-one game engine solution, OGRE can be suitable choice.

- **Simple Direct Media Layer, SDL:** SDL is a cross-platform multimedia library, developed with commercial game porting in mind (in fact, it has already been used to port a number of games from Windows to Linux, including most of Loki's titles). SDL supports almost all major operating systems, including Linux, Windows, BeOS, and MacOS. In addition to fast graphics support, SDL provides interfaces for playing sound, accessing CD-ROM drives, and portable multithreading. SDL is also an excellent library for free software projects: Released under the GNU LGPL, it has everything a programmer needs to write fast, portable games. SDL has accumulated a collection of add-on libraries which provide additional functionality for game developers. SDL's website is <http://www.libsdl.org>.
- **OpenGL:** OpenGL is a 3D graphics API designed by Silicon Graphics. Though not originally intended as a game programming library, OpenGL has found a place as a convenient interface standard for hardware-accelerated 3D graphics, and therefore lends itself well to gaming. The Mesa 3D Graphics Library is a free implementation of the OpenGL specification, and there are Mesa-based Linux drivers for several popular 3D accelerator cards. Unfortunately we can't cover OpenGL in the detail it deserves (3D graphics is a subject of its own and not specific to Linux programming), but we'll at least demonstrate how to gain access to OpenGL from within SDL programs. This particular combination allows us to use the rendering power of hardware-accelerated OpenGL with the various amenities provided by SDL, and it is an excellent platform for developing games. Loki Software has

successfully used SDL and OpenGL to port several commercial games to Linux, including Heavy Gear II and Soldier of Fortune.

- **Glide:** Glide is 3Dfx's native 3D programming library, designed specifically for 3Dfxgraphics chips. It is a much lower-level library than OpenGL, serving mainly as a consistent interface for all video cards based on 3Dfx chipsets. Since 3Dfx no longer has a virtual monopoly in the 3D accelerator business, Glide has lost a certain amount of popularity recently. With the advent of accelerated OpenGL under Linux, there are very few good reasons to use Glide for new game projects. More information about Glide is available at <http://www.3dfx.com>.

2.2.2 Sound:

Because of importance of sound, we think to use a sound library which is free license, 3D audio suitable for a football game. Some sound libraries are here:

- **Advanced Linux Sound Architecture, ALSA:** ALSA is a community project that seeks to surpass OSS in all areas. The ALSA team has created a complete set of kernel-level sound card drivers, an easy to use programming interface, and a facility for emulating OSS. ALSA is not without its fair share of quirks, but it is a viable alternative to OSS for sound support and, with few exceptions, games that support OSS are also compatible with ALSA. It would be good to see ALSA grow in popularity since it has a lot of functionality and a lot of promise. The only serious problem with ALSA is that it is somewhat of a moving target; its API changes frequently.

- **The Open Audio Library, OpenAL:** OpenAL is an environmental 3D audio library that supports just about every major platform. It aims to provide an open replacement for proprietary (and generally incompatible) 3D audio systems such as EAX and A3D. OpenAL can add realism to a game by simulating attenuation (degradation of sound over distance), the Doppler effect (change in frequency as a result of motion), and material densities. OpenAL has been used in several Linux game ports, including Heavy Gear II and Sid Meier's Alpha Centauri.

2.2.3 Artificial Intelligence:

We think to use a basic AI for our game when playing against computer and also for goalkeeper. In games, AI can be divided into two categories: deterministic and non deterministic AI. In determine AI there is no suspicion. Deterministic behavior is specified and predictable. Deterministic AI techniques are the easiest part of AI. They are fast, easy to implement, test, debug predictable. Deterministic AI does not make easy learning and evolving. Also it is predictable after a little game play. Nondeterministic AI is related about learning and unpredictability. Since we plan using a primitive AI, it will not be nondeterministic.

2.2.4 Network:

Networked gaming is big, and it is here to stay. There are several networking interfaces for Linux, but almost all of them revolve around the BSD sockets API that became a standard part of UNIX years ago. In our game, users from different computers can play a game together from internet or a same lan. Some APIs are:

- **IPX and SPX:** IPX (Internetwork Packet eXchange) is a simple networking protocol similar to the Internet's underlying IP protocol, and SPX (Sequenced Packet eXchange) is a higher-level protocol similar to the Internet's TCP protocol. These protocols (often collectively referred to as IPX) were designed by Novell for its NetWare line of products. IPX has fallen out of favor, but it is still used in a number of games. IPX is fine for small private LANs, but it is not ideal for large networks should you choose to support IPX in your games, the Linux kernel provides the necessary networking code (via the normal BSD sockets interface). It is not terribly difficult to support both TCP/IP and IPX with the same networking code.
- **BSD Sockets:** A socket is a UNIX file descriptor that designates a network connection rather than a file on disk. Sockets can be thought of as telephone handsets; they are communication endpoints, through which data can be transferred in either direction. Sockets are most commonly used with TCP/IP, the stack of protocols behind the Internet. The advantage of programming with TCP/IP sockets is that TCP/IP is an incredibly versatile protocol. Some version of the sockets API can be found in nearly every operating system, including Linux, Windows, BeOS, and MacOS TCP/IP can be used for both local (LAN) and wide-area (WAN) networking, and the protocol can be adapted to the nature of a particular game.

2.3 Game Engine:

Since game needs some calculations, the function game engine here is computing the movement of ball according to user input and physics law. It defines what is going on next when a collision happens between ball and players

or poles or walls etc. Some game modes like “street football” need to calculating movement of ball, so we think to use as a game engine;

- Open Dynamics engine
- Physics
- Opal
- Newton Game Dynamics

3 DESCRIPTION

3.1 Game Modes

In this mode game modes will be explained. The properties of game modes are listed below:

3.1.1 Traditional Mode

- The number of players are 22. Each team has 11 players.
- There will be two half; each half will be equal.
- Each team will have minimum 7 maximum 3 substituons.
- Match lenght will be between 5 and 30 minutes.
- There will be a referee.
- There will be rules like penalty, throw in, corner, goal kick.

- Games played at classical stadiums.

3.1.2 Japanese Goal Mode

- In this mode there will be no goal keeper.
- Size of a goal will become smaller.
- Each team will have 2 or 3 players.
- No referee.
- There will be specific rules for Japanese mode.
- If same team has 3 corners there will be penalty shoot out
- There is no throw in or goal kick.

3.1.3 Street Football Mode

- Include 1 vs 1 or 2 vs 2 playing.
- There will be only one goal and one goalkeeper.
- There is no game time.
- There will be a goal limit to win the match, for example; 3 goals win.

3.1.4 For Fun Modes

3.1.4.1 German Goal Mode

- There will be three players.

- There will be specific rules for this mode.
- The game will not played in a stadium.
- There will be different place choices.

3.1.4.2 Indoor Mode (Saloon Football)

- The number of players is 12; each team has 6 players.
- Matches will be palyed on 2 half.
- No substituton.
- No rules such as; throw in, goal kick, corner kick.
- No referee.
- There are different indoor stadium choices.

3.1.5 Training Mode

- User skills can be improved by doing exercise.
- There will be a training menu for user. In this menu following items are exist
 - Shooting
 - Penalty Shoot out

- Corner kick
 - Free kick
- Different training places can be chosen; stadiums, streets, indoor places.
- League Mode
- Different league choices such as; seri a, la liga.
- Each league choice has specific league cups.
- Each league has certain number of matches.

3.1.6 Edit Mode

- Player can create their own players.
- Physical attributes can be chosen by player.
- Football skills can be chosen by player.
- Every created footballer will be saved to database.

3.1.6.1 Footballer Properties

- Every football player has unique attributes
 - Abilities

- Age
 - Condition
 - Injury
 - Shooting
 - Current team
 - Height
 - Constituon
 - Agility
 - Mentality
 - Reputition
-
- The values of these properties are between 0 and 20.
 - According to these properties, each football player will have a power between 0 and 20.

3.1.6.2 Team Properties

- Each team have unique properties;
 - Offense

- Deffense
 - Speed
 - Strength
 - Overall point
- Properties are between 0 and 20.
- Overall properties will be decided according to avarage of other properties.

3.2 Weather Condition

- There are three main weather conditions; fine, rainy, snow.
- Each weather conditions have different effects on outdoor football environment.
- Each weather condition will have different graphic properties.

3.3 Enviroment Properties

- Outdoor playing environments.

- For this mode environment is streets, gardens etc.
- Various of environments will be in game.
- Stadium.
- Traditional mode; game will be played in classic stadium.
- Various of stadiums will be in game.

3.4 Sound Properties

- Goal celebrations
- Tackle sound
- Whistle sound
- Ball hit sound
- Audience sound

4 REQUIREMENTS

4.1 Functional Requirements

4.1.1 Menu Requirements

4.1.1.1 Start Menu

Friendly Game:

This menu option will open a sub menu. In this sub menu player will be able to choose a league, a team to play from that league and likewise another team to play with. After selecting these, player will be able to start the match.

Start League:

This menu option will enable the player to play a whole league matches. By clicking this menu a sub menu shall open. In this sub menu player will be able to choose a league and a team. Afterwards league mode will be opened.

Continue League:

This menu option will enable player to continue a saved league. After clicking this menu item, a list of saved games will be shown and player will be able to select desired record from list.

Multiplayer Game:

By clicking this menu item, a sub menu will open. Using the choices in this sub menu, player will be able choose a multiplayer game or an already open multiplayer game.

Fun Modes:

This menu item will open a sub menu. From this sub menu, player will be able to choose between “Japanese Goal Match” , “Street Football” and “German Goal Match” fun modes. By selecting any of these options another sub menu which asking to choose league and team will be opened.

Training Mode:

This menu item will open a sub menu to choose between “Shoot”, “Free Kick”, “Penalty” or “Corner Kick” modes. Choosing these will open restricted game modes to train.

Exit:

This menu item will close the game and send player to operating system window.

4.1.1.2 In Game Menu

Resume:

This choice will return game play.

Substitution:

This choice will open the sub menu to make substitutions.

Tactics:

This choice will open sub menu to change team tactics.

Exit:

This choice will end match to previous menu

Exit to Operating System:

This choice will close the game and send player to operating system window.

4.1.2 In Game Functions

Passing:

This will send the ball to another player. Going of ball graphics will be seen and sound of the kick will be heard. Also audience sound will change if shoot is successful.

Shooting:

Ball will be sent through the goal. Again ball graphics will be seen and sound of kick will be heard.

Dribbling:

Selected character will start running faster. Audience sound will change if this is an offensive move.

Intentional Foul:

Opponent player will be seen to fall. Audience sound will change and referee sound will be heard.

Tackling:

Selected character will be seen sliding. If tackling is successful audience sound will change.

4.2 Structural Requirements

4.2.1 Graphics

Graphics is the main requirement for a game project. In a football game; players, ball, stadium and other visual aspects require high 3D graphics. Our goal in graphics aspect of the project is to give players a realistic game playability. To maintain this this goal, a graphics engine shall be used.

4.2.2 Sound

Sound is another important aspect of the project. For reality issues and to make players feel in game; player, audience, referee, announcer, environment sound shall be added to the game. To achieve this, a sound engine shall be used.

4.2.3 Network

To maintain multiplayer gaming between different computers, network requirements shall be satisfied.

4.2.4 Artificial Intelligence (AI)

Both single player and multiplayer parts of the game requires artificial intelligence. For multiplaying, movements of non selected characters shall be done with artificial intelligence. For single playing, in addition to multiplaying; tactics of the opponent team, decisions of tackling, passing or shooting shall be done with artificial intelligence, too.

4.2.5 Physics

Physics has second priority on this project's requirements. It shall be applied only to ball movements.

4.3 Performance Requirements

- Artificial Intelligence has to make decisions quick enough for fluent game playability.
- Network connection has to ensure that game won't be interrupted during game playing.

4.4 Software Requirements

A Linux Operating System and related libraries will be required.

4.5 Hardware Requirements

These are required for game play:

- Graphics card supporting 3D graphics
- Sound Card
- Network Connection

4.6 Non Functional Requirements

4.6.1 Security

Game will have network playability. Thus, there shall be security measurements of protecting data from outside machines.

4.6.2 Maintainability

Project will be open source. This feature will provide maintainability by other developers even project group stops working on it.

4.6.3 Portability

Game shall be portable through linux systems.

4.7 Project Constraints

4.7.1 Time Constraints

Deadline of our project is May 2010. All, design, implementation, testing and adjustments must strictly meet this deadline.

4.7.2 Language Constraints

All of our group members know using C++. Thus, the language for the project will be C++. Platform independency and code portability is an implementation constraint. Development environment will be GCC based environment for Linux/Unix.

4.7.3 Data Constraints

A fair amount of primary storage is required to hold various data structures for analyzing data flow over the network, if the user chooses to save some data for later.

4.7.4 User Interface Constraints

The use interface has to be simple and easy to understand. The menus and GUI elements will be in the interface.

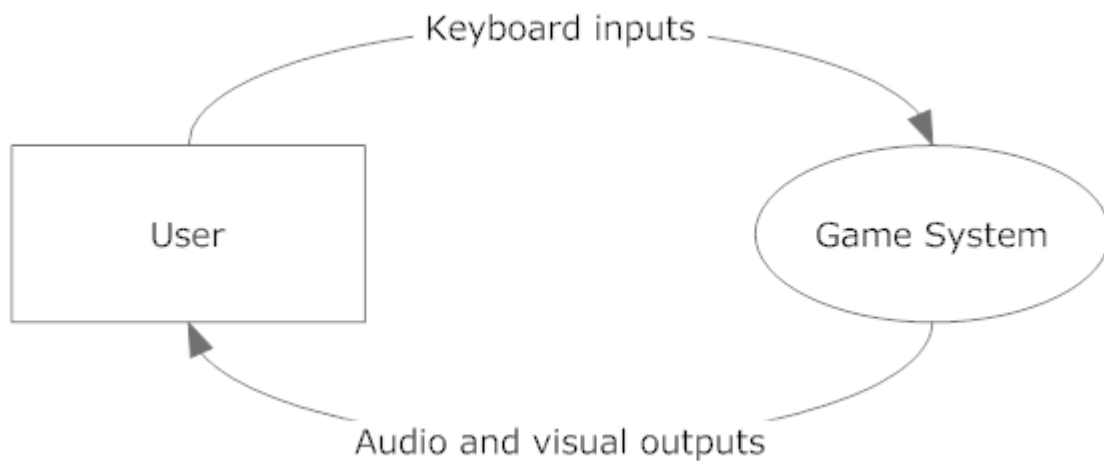
5 SYSTEM ANALYSIS AND MODELING

5.1 Functional Modeling

5.1.1 Level 0 of Data Flow Diagram

Level 0 of data flow diagram shows the interactions between the user and game

system. User initiates control inputs and and game systems responds with visual and audio outputs.

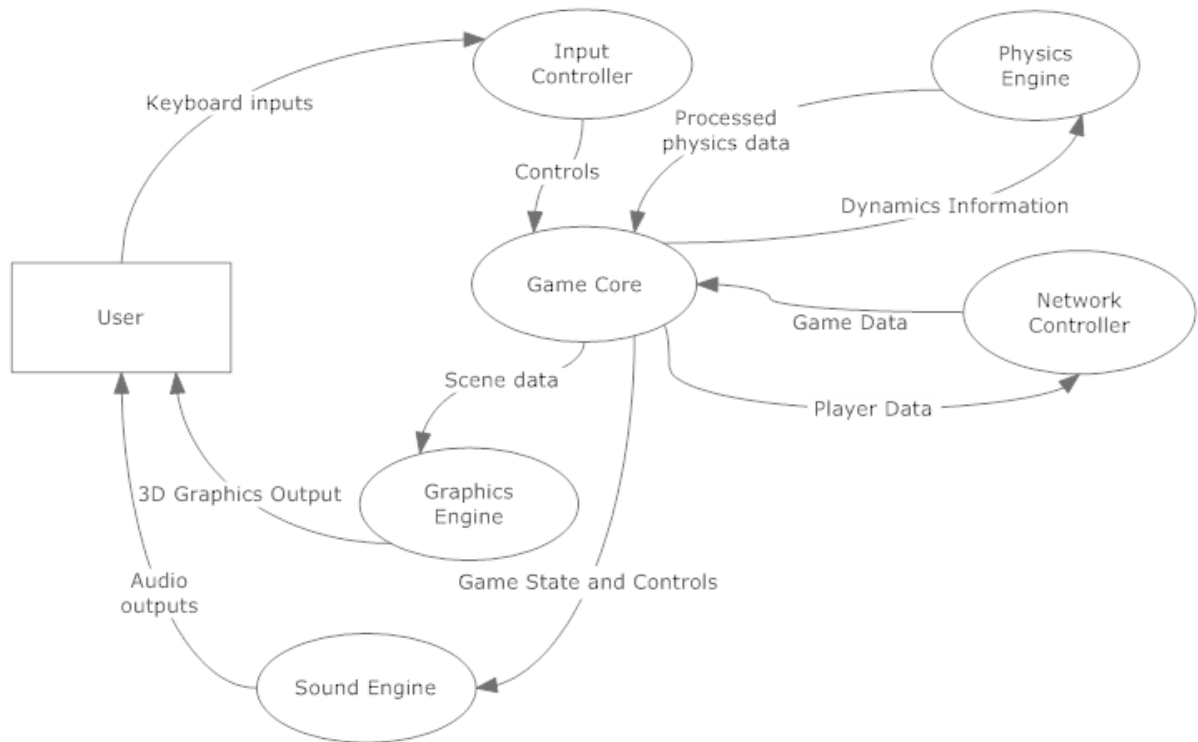


5.1.2 Level 1 of Data Flow Diagram

Level 1 of data flow diagram shows the details of the game system, describing the relations and interactions between its main components such as graphics engine, sound engine, physics engine etc.

Requirements Analysis Report

Geeks in Action

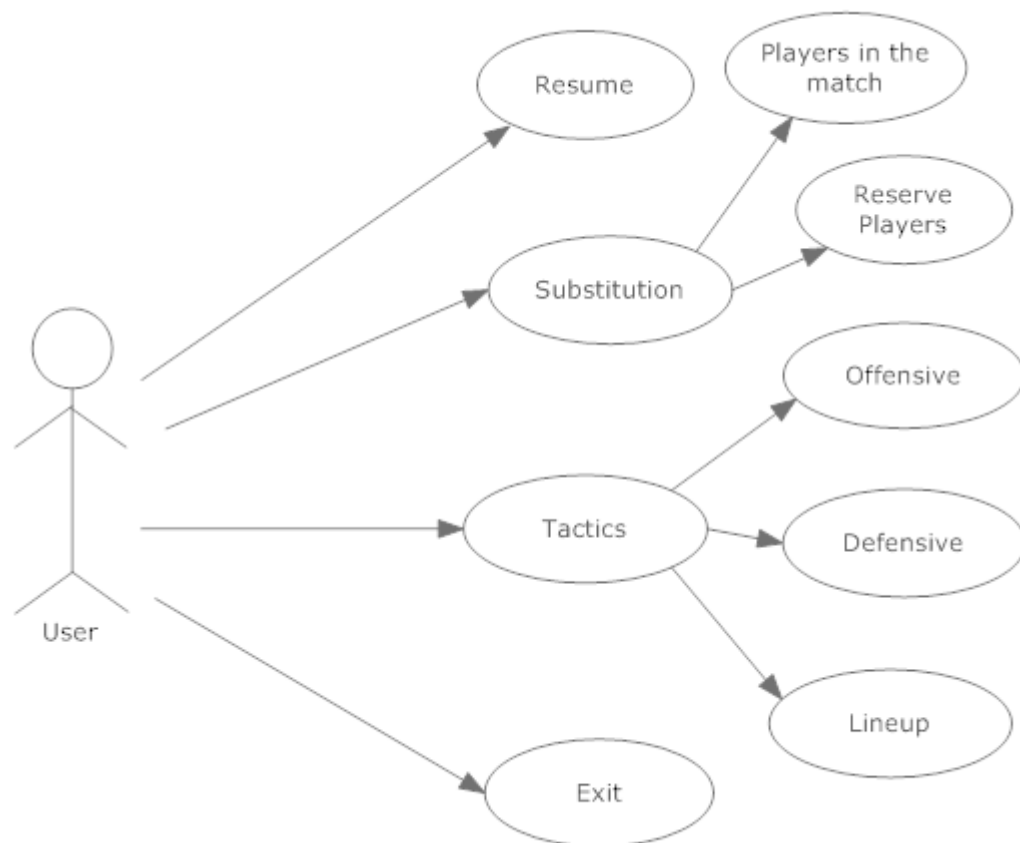


5.2 Use Case Analysis

5.2.1 Start Menu Use Case

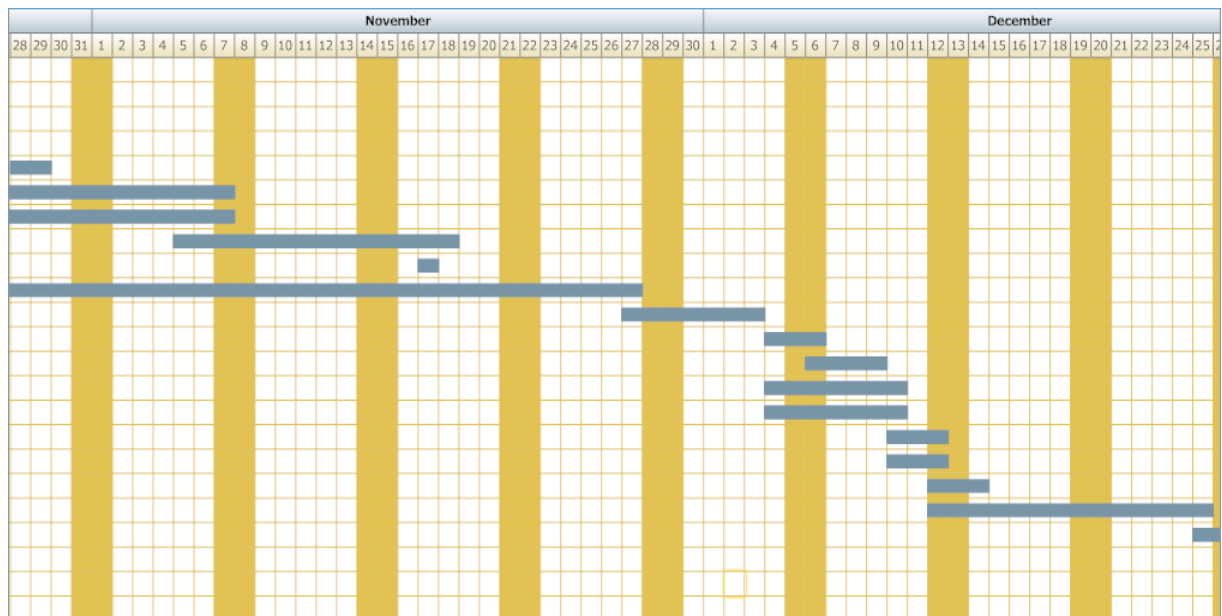


5.2.2 In Game Menu Use Case



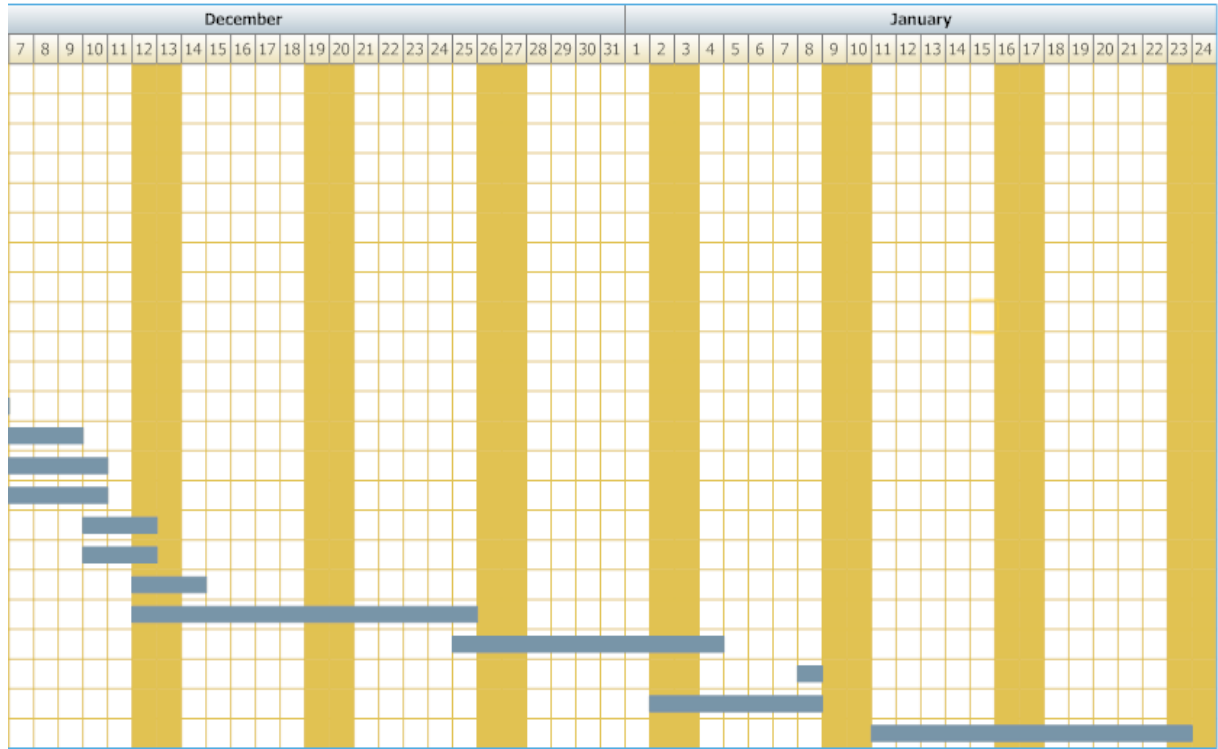
Geeks in Action

Number	Task	Start	End	Duration	September			October																							
					28	29	30	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	Choosing the project	28/9/2009	13/10/2009	12																											
2	Understanding the project	13/10/2009	17/10/2009	4																											
3	Project Proposal	17/10/2009	26/10/2009	6																											
4	Submission of Project Proposal	26/10/2009	26/10/2009	1																											
5	Technical Meeting	28/10/2009	29/10/2009	2																											
6	Software Research	26/10/2009	7/11/2009	10																											
7	Complimentary Material Research	26/10/2009	7/11/2009	10																											
8	Requirement Analysis Report	5/11/2009	18/11/2009	10																											
9	Submission of Requirement Analysis Report	17/11/2009	17/11/2009	1																											
10	Further Research on Software	17/10/2009	27/11/2009	30																											
11	Holiday	27/11/2009	3/12/2009	5																											
12	Conclusion and Installation of Software	4/12/2009	6/12/2009	1																											
13	Experimentation of Software	6/12/2009	9/12/2009	3																											
14	Designing graphical user interface	4/12/2009	10/12/2009	5																											
15	Detailed modularization of the system	4/12/2009	10/12/2009	5																											
16	Design of sub modules	10/12/2009	12/12/2009	2																											
17	Initial Design Report	10/12/2009	12/12/2009	2																											
18	Submission of Initial Design Report	12/12/2009	14/12/2009	1																											
19	Detailed design of graphical user interface	12/12/2009	25/12/2009	10																											
20	Detailed design of sub modules	25/12/2009	4/1/2010	7																											
21	Submission of detailed design report	8/1/2010	8/1/2010	1																											
22	Team Presentation	2/1/2010	8/1/2010	5																											
23	Prototype Demo	11/1/2010	23/1/2010	10																											



Requirements Analysis Report

Geeks in Action



Software Research: Talat Özer, Oğuz Şen

Complementary Material Research: Cuma Kılınç, Nur Muhammet Arınç

Designing Graphical User Interface: Talat Özer, Oğuz Şen

Design of sub Modules: Cuma Kılınç, Nur Muhammet Arınç

Rest of the subjects are assigned to whole team.