Software Requirements Specification for Triangulation Games

Version 1.3 not approved

Prepared by:

**Ville Parviainen,
Lauri Kainulainen,
Suvi Peltomäki,
Marianne Haugen,
Jon Sahlberg,
Salvador Jesús Romero Castellano,
Umair Azfar Khan,
Kyösti Karila**

4.12.2005

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Ville Parviainen	22.10	Added the introduction chapter	0.1
Lauri Kainulainen	23.10	Added the chapter 2	0.2
Ville Parviainen, Lauri Kainulainen	23.10	Wrote chapters 3,4,5 and appendix B	0.5
Ville Parviainen	26.10	Finishing chapter 1	0.6
Lauri Kainulainen	30.10	Review. Rewrote section 1.4	0.7
Lauri Kainulainen	31.10	Added Appendix B and chapter 7	0.8
Lauri Kainulainen, Ville Parviainen Ville Parviainen, Umair Azfar Khan, Salvador Jesús Romero Castellano	9.11 4.12	Fixed things pointed out in the review, added tags for non-functional requirements Use cases added along with use case diagrams. Added Appendix C.	1.0 1.1

1 Introduction

1.1 Purpose

This document describes the requirements for the Triangulation Games software.

1.2 Document Conventions

All the references in this document have been made by using running numeration. When you see a number between closures, example [1] this means that it refers to the first reference mentioned in the reference section at the end of this document.

There is a glossary at the end of the document, where the more uncommon terms are explained.

1.3 Intended Audience and Reading Suggestions

The document is meant for the software project group and also for the client to specify what will be implemented in the Triangulation Games software.

1.4 Project Scope

Triangulation Games software is an application that makes it possible to play various different combinatorial games as solitaire, against the computer's AI or with another player. The system also functions as a platform for defining new types of triangulation games.

2 Overall Description

2.1 Product Perspective

Triangular Games is a stand-alone software application that is meant to realize games described in the article "Games on Triangulations" [1].

2.2 Product Features

The software makes it possible to play multiple triangulation games (see [1], for more information on triangular games). The games may be played alone or against another person, depending on the type of game. A graphical user interface will be shown to the user. This user interface is mainly mouse controlled but may also be used with a keyboard as well.

By default, the system will be able to implement three different games, namely constructing, transforming and marking. New games may be added to the software easily without the need to modify the code of the original application. These added games will be automatically selectable through the application's graphical user interface.

2.3 User Classes and Characteristics

The main user class is player. This is the most important user class and the user interface design will be geared towards serving the player. The software is however designed mostly for research oriented people coming from areas such as mathematics, social sciences and computer sciences.

Although all research people have an academic background, the technical level of experience may vary greatly. This means that no assumptions about highly experienced users can be made. The user interface will be designed to serve the needs of inexperienced players.

Since adding new games to the software will also be possible, one user class is the game developer. People from this user class will have experience in playing the games and the scientific background behind creating new games. This user class is not as important as the player class and users of this class are mostly interested in the ease of defining new games.

2.4 Operating Environment

The software will be implemented using the Java-platform. This makes it possible to run the application on multiple different environments without any modification to the code. The software requires a mouse or a keyboard, a graphical user environment (X window system, Windows, Mac, etc.), and Java-runtime environment (version 1.4 or later)[2].

2.5 Design and Implementation Constraints

The client has requested the use of the Java environment, so the platform has been chosen before design. Because of the cross-platform requirement, the software cannot be dependent on platform-specific libraries.

The software should run well on any computer which has Java 2 (version 1.4 or later) Runtime Environment (JRE)[3] installed. Installation should be easy, so no external databases should be required.

2.6 User Documentation

Documentation about how to define new games will be provided to make it possible for non-developers to create new games for the system. There are going to be two end user documentations:

- User manual, for end users. It will describe how to use the system for playing.
- Maintenance manual, for end users and new developers. It will describe how to specify new games for the system. It will also contain the design documents and interfaces of the system, as well as the source code. If necessary, it will describe also how to implement a plug-in (such as an AI) for the system.

2.7 Assumptions and Dependencies

This software will be released under the GPL-license.

3 System Features

3.1 System has a graphical user interface

3.1.1 Description and Priority

The system has an easy to use graphical user interface which supports both a mouse and a keyboard.

This feature is essential for the system.

3.1.2 Use Case

UC - 001	The user starts the application	V 1.0
Description	This is a system initiation process.	
Pre-condition	The System has been correctly installed in the computer.	
Post-condition	The System is running and shows a graphical interface to the user. The user is able to use the system.	
Normal Sequence	The user activates the application. The System starts and shows the graphical interface to the user. The user is able to use the system.	
Exceptions	1.	Condition: There is a problem in loading a game or an artificial intelligence.
		Action: The system informs the user about the error.
		Ending: The program is terminated.

3.1.3 Functional Requirements

GUI	The system uses a graphical user interface to display data to the user
GUI.keyboardInput	The system can be used fully with a keyboard
GUI.mouseInput	The system can be used fully with a mouse

3.2 Player can choose multiple game types

3.2.1 Description and Priority

The system supports multiple games, and new games can be added by the end user. By default, the system will implement three games. The priorities of the triangulation games are described in the following table:

Table 1: Game implementation requirements

Category	Game Type	Priority
Constructing (a Triangulation)		
	Monochromatic complete triangulation	1
	Monochromatic triangle	3
	Bichromatic complete triangulation	2
	Bichromatic triangle	3
Transforming (a Triangulation)		
	Monochromatic flipping	1
	Monochromatic flipping to triangle	3
	Bichromatic flipping	3
	All-green solitaire (monochromatic)	2
	Green wins solitaire	3
Marking (a Triangulation)		
	Triangulation coloring game (monochromatic)	1
	Bichromatic coloring game	2
	Four-cycle game (monochromatic)	4
	Nimstring game (monochromatic)	4

This feature is of high priority.

3.2.2 Use Case

UC - 002	User starts a new game	V 1.0
Description:	This use case describes the steps that are required to start a new game.	
Precondition:	The System is running and is not busy.	
Post-condition:	A new game, a starting position and an artificial intelligence has been selected in the game menu.	
Normal Sequence:	The user starts a new game with the menu. The system shows a list of games available. The user selects one of these games. The system offers the user the possibility of select a starting position. If the user accepts, the UC-003 is executed. Otherwise, a random starting position is selected. The system offers the user the possibility of select an artificial intelligence for each player of the game. If the user accepts, the UC-004 is executed. If not, human player is selected for the first player, and the random artificial intelligence is selected for the second. The system loads the selected game.	

Exceptions:	1.	Condition: There is no game available.
		Action: The System shows an empty list. No possibility except canceling to run a game.
		Ending: The use case terminates.

3.2.3 Functional Requirements

Game.Multiple	The system supports multiple games, and can load additional games from an external source

3.3 Player can choose from multiple opening positions

3.3.1 Description and Priority

After selecting the game the player selects the opening position for the game. Different types of opening positions are available for different game categories. Further information about this can be found in the article "Games on Triangulations" [1]. The player can also start the game from a random position.

This feature is of high priority.

3.3.2 Use Case

UC - 003	The user selects an opening starting position with the menu	V 1.0
Description:	The user is required to select an opening position from the menu before starting any game.	
Precondition:	A game has been selected.	
Post-condition:	The Game starts with the selected opening position.	
Normal Sequence:	The user starts a new game. The System displays a list with the opening position available, and the random opening position. The Game User selects one of them. The new opening position is selected as current opening position.	

3.3.3 Functional Requirements

Game.MultipleOpeningPositions	The system has support for multiple opening positions that can be applied to all the games.

3.4 Players can be both a Human or a Computer

3.4.1 Description and Priority

When starting a game, the user must be able to select in between Human or Artificial Intelligence for playing a game.
This feature is of a medium priority.

3.4.2 Use Cases

UC - 004	The System requires the user to select the nature of a player in the current game.	V 1.0
Description:	The user selects the nature of the players for the current game so that the game can be played by having both players as human or both players as computers or one player computer and the other human.	
Precondition:	-	
Post-condition:	The nature of the players has been changed.	
Normal Sequence:	The System offers the Game User to select between a human player or an artificial intelligence. If the user selects artificial intelligence, the UC-005 is executed. If not, human is selected as nature of the player.	

3.4.3 Functional Requirements

Game.TwoPlayerSupport	The system shall support two player turn-based gaming.

3.5 There must be a default random artificial intelligence

3.5.1 Description and Priority

A random artificial intelligence plays a game by choosing at random one of the legal movements at any moment in the game. Having a random artificial intelligence is must for the system.

This feature is of high priority.

3.5.2 Use Case

UC - 005	The user selects an artificial intelligence		V 1.0
Description:	The system must always provide the artificial intelligence that can play using random moves.		
Precondition:	-		
Post-condition:	An artificial intelligence has been selected.		
Normal Sequence:	The System offers the user a list with all the different artificial intelligences available. The Game User selects one from the list.		
Exceptions:	1.	Condition: No external artificial intelligence could be found.	

		Action: The random artificial intelligence is always available. It will be the only option of the list.
		Ending: The user selects the random AI.

3.5.3 Functional Requirements

AI	The system has an artificial intelligence that the player may play against. The artificial intelligence supports every two player game.
Game.TwoPlayerSupport	The system shall support two player turn-based gaming.

3.6 The user can change the nature of players during a game

3.6.1 Description and Priority

While playing, the user can change the nature of the players in the game, without the need of a restart. The game continues with the new selection of players.

This feature has a medium priority.

3.6.2 Use Case

UC - 006	The user changes the nature of the player during a game. V 1.0
Description:	If during the game a player wishes to leave, there should be the possibility to assign an AI to take over from that position.
Precondition:	A game is already running.
Post-condition:	The nature of the player has been changed.
Normal Sequence:	The user wants to change the nature of the players in the current game with a menu. The System shows a list of all the players that can support the substitution. One of the players is selected. UC-005 is executed.
Comments:	2. In a solitaire game, only one player can be changed. In a two player game, both players can be changed.

3.6.3 Functional Requirements

Game.ChangePlayer	The system has an embedded AI that takes over from a player when asked to do so.

3.7 New Artificial Intelligence can be loaded from a file

3.7.1 Description and Priority

A third person must be able to develop and add new AI to the system. The user will then be able to select from them as players while playing a game.
This feature is of a low priority

3.7.2 Use Case

UC - 007	The AI Developer develop and add a new AI V 1.0
Description:	All the new artificial intelligences will be placed in their respective files. The system will load these files and show the available artificial intelligences when the user is selecting the types of player.
Precondition:	-
Post-condition:	A new AI has been added.
Normal Sequence:	The AI Developer designs a new AI for a game. The AI Developer installs the AI in the System, as described in the manual. The user starts a new game. The system loads the new AI from the implemented file.
Comments:	1. This Use Case is a general one with a high level of abstraction. 2. See UC-002 and UC-006 for more details

3.7.3 Functional Requirement

Game.LoadAI	The system has files containing implementation of different artificial intelligences that the system can load.
Game.AIInterface	The system prompts the user about implementing the new AI's

3.8 The games are defined separately from the software code

3.8.1 Description and Priority

All the games in the system are separately defined in an external source. The definition for a game has to be simple and understandable by a non-programmer. This can be achieved for example with an XML file or a text document. The file has to contain certain attributes, such as the game type, the ending conditions and the starting point or points. The games can be loaded from a predefined directory or runtime with a load file dialog.

This feature is essential and has a high priority.

3.8.2 Use Case

UC - 008	A new game is specified in a file V 1.0
Description:	The system loads all the different types of games from separate files.
Precondition:	-
Post-condition:	A new game has been added
Normal Sequence:	The user defines a new game in a file. The file is placed within the system to be used. The System loads the new game from the file.
Comments:	1. This Use Case is a general one with a high level of abstraction. 2. See UC-002, UC-003, UC-007 and UC-010 for more details
Exception:	Condition: The necessary attributes are not found or are not defined properly. Action: The system informs the user about the problem Ending: The use case aborts

3.8.3 Functional Requirements

Game.Defined	The system supports defining new games without modifying the source code.
Game.AutomaticSearch	The system searches an external source for games on startup and loads them.

3.9 The games end on a predefined condition

3.9.1 Description and Priority

Every game has an ending condition, defined in the corresponding game definition. When this condition is carried out, the game ends. The system then proceeds to calculate the winner and shows the points for each player.

This feature has a high priority.

3.9.2 Use Case

UC - 009	All the games end on a pre-defined condition V 1.0
Description:	When the winning condition of a game is achieved, the game ends and the scores of the players are displayed.
Precondition:	A game is being played.
Post-condition:	The current game is finished and is not playable anymore.
Normal Sequence:	One player makes a move that carries out the ending condition, or is the last possible. The system reports the user about the end of the game. The Game User can not play the game anymore.

Exception:	1. If the ending condition has not been defined, the game may never end.

3.9.3 Functional Requirements

Game.End	All the games end on a pre-defined condition

3.10 Game saving feature

3.10.1 Description and Priority

The user will be able to save the game and continue when the game is loaded.

This feature has a low priority.

3.10.2 Use Case

UC - 010	The Game User saves a game V 1.0
Description:	The system must have the capability to save the current state of the game being played.
Precondition:	A game is currently being played
Post-condition:	The game has been saved in a file.
Normal Sequence:	The user selects the save option from the menu. The system opens a dialog window where the user can specify the game's name and the location where it can be saved to. The user makes the necessary choices. The System saves the game in a file.

UC - 011	The Game User loads a game V 1.0	
Description:	The system must be able to load a game from a file.	
Precondition:	A game has already been saved to a file.	
Post-condition:	The game has been loaded from a file, and is currently being played.	
Normal Sequence:	The user selects the load option from the menu. The system opens a dialog window for the user to choose a file to load. The user makes the necessary choices. The System loads the game from the file.	
Exceptions:	1.	Condition: No saved file already exists.
		Action: The user clicks on cancel.
		Ending: The use case aborts.

3.10.3 Functional Requirements

Game.Save	The system saves the game in a file.
Game.Load	The system loads the game from a file.

3.11 The in-game help function

3.11.1 Description and Priority

While playing, the user is able to search the in-game help file for solutions to basic problems regarding the game rules and functionalities.

This feature has a medium priority.

3.11.2 Use Case

UC - 011	The user locates a solution to a problem with the help function. V 1.0
Description:	The help function makes it possible for a user to search for solutions to basic problems related to the different aspects of the triangulation game.
Precondition:	The user has a problem with starting the games or with the rules of the games.
Post-condition:	The problem has been solved.
Normal Sequence:	The user wants to search the help file for a solution or a hint. The System displays the contents of the help file, and the user chooses the appropriate one.

3.11.3 Functional Requirements

Game.Help	The system has an embedded help file that contains basic solutions and hints for problematic situations

4 External Interface Requirements

4.1 User Interfaces

UI-1: All the user interfaces will be designed following common guidelines in the usability document.

UI-2: The user interface uses one main window for the actual game playing and dialog windows for changing the general settings of the system.

4.2 Hardware Interfaces

HI-1: The system functions on a computer that has a screen for output and keyboard or mouse or both for input.

4.3 Software Interfaces

SI-1: The system requires a working Java-environment [2] and a graphical user interface supported by the Java Swing library [4].

5 Other Nonfunctional Requirements

5.1 Performance Requirements

PR-1: The system should be executed without problems in a machine that fulfills the requirements of the Java runtime environment [2].

PR-2: The basic artificial intelligence must operate fast enough. The next move should be calculated within 10 seconds.

5.2 Safety Requirements

The system does not use or receive network connections and is used locally. This means that no network safety requirements exist. Overwriting an existing file on the file system has to be confirmed from the user to prevent accidental loss of data.

5.3 Security Requirements

No personal information is stored by the system so no specific security requirements need to be defined.

5.4 Software Quality Attributes

The system is playable with the minimum setup defined earlier in Hardware Requirements.

SQA-1: The system shall run well on both Linux and Windows and should be playable with the mouse or the keyboard or both.

6 Traceability

Most of the requirements in this document are based on the initial interviews with

Timo Poranen (the client) and brainstorming meetings of project group. The article [1] provided specific and exact requirements that are directly related to the nature of triangular games.

7 State of the Requirements

7.1 Stable Requirements

All the requirements having medium or high priority are stable requirements and will not undergo any change.

7.2 Volatile requirements

The requirements with low priority might go under change as the system takes shape and implementing them might prove to be a waste of time and resources.
References

This document refers to the article Games on triangulations, which specifies the theoretical basis for the triangulation games.

References

[1] Games on triangulations, Oswin Aichholzer, David Bremner, et al., Theoretical Computer Science 343 (2005) 42-17 (available online at www.sciencedirect.com)

[2] Java platform, http://java.sun.com

[3] GNU General Public License, GNU.org webpage: http://www.gnu.org/copyleft/gpl.html

[4] The Swing component set is part of a new class library called the Java Foundation Classes, or JFC. Swing, is a new GUI component kit that simplifies and streamlines the development of windowing components. Windowing components are the visual components (such as menus, tool bars, dialogs and the like) that are used in graphically based applets and applications. http://java.sun.com/products/jfc/tsc/articles/getting_started/index.html

Appendix A: Glossary

AI	Acronym of “artificial intelligence”. In this context an Artificial Intelligence is a software entity that is able to calculate and decide movements for an instance of a game while it is being played, thus making it able to play games on its own.
Developer	A person involved in the code generation of the system. There are both those belonging to the actual project group, and those who want to improve the system once it has been published under GPL license.
Ending Condition	A condition described in the rules of a game that defines when a game ends.
End User	People who will execute and use the system. There are both those who are going to play, or those who are going to develop new games, starting positions and/or artificial intelligences.
Game	A triangulation game defined in the article [1]
GPL	General Public License. See [3]
Graph	See [1]
Opening position	The starting position of a game. Coordinates for the point which defines the graph. Details in [1].
Player	In the game context, the entity that makes a movement in a turn. While playing, a person or an artificial intelligence machine can both play the role of a player. Since a player can be also an AI, it’s necessary not to confuse a ‘player’ (concept) with an ‘end user’ (people).
Random Opening Position	An opening position made by the system with random starting parameters.
System	The triangular game application that this document describes.

Appendix B: Requirements List

Functional Requirements

AI	The system has an artificial intelligence that the player may play against. The artificial intelligence supports every two player game
Game.AutomaticSearch	The system searches an external source for games on startup and loads them.
Game.Continue	The system has an embedded AI that takes over from a player when asked to do so.
Game.Defined	The system supports defining new games without modifying the source code.
Game.End	All the games end on a pre-defined condition
Game.Help	The system has an embedded help file that contains basic solutions and hints for problematic situations
Game.Load	The system loads the game from a file
Game.Multiple	The system supports multiple games, and can load them from an external source.
Game.MultipleOpeningPositions	The system has support for multiple opening positions that can be applied to all the games.
Game.Save	The system saves the game in a file
Game.SaveOpeningPositions	The system saves all the defined opening positions so that they are available after a restart.
Game.Solitaire	The system has support for solitaire games
Game.TwoPlayerSupport	The system shall support two player turn-based gaming.
GUI	The system uses a graphical user interface to display data to the user

GUI.keyboardInput	The system can be used fully with a keyboard
GUI.mouseInput	The system can be used fully with a mouse

Non-Functional Requirements

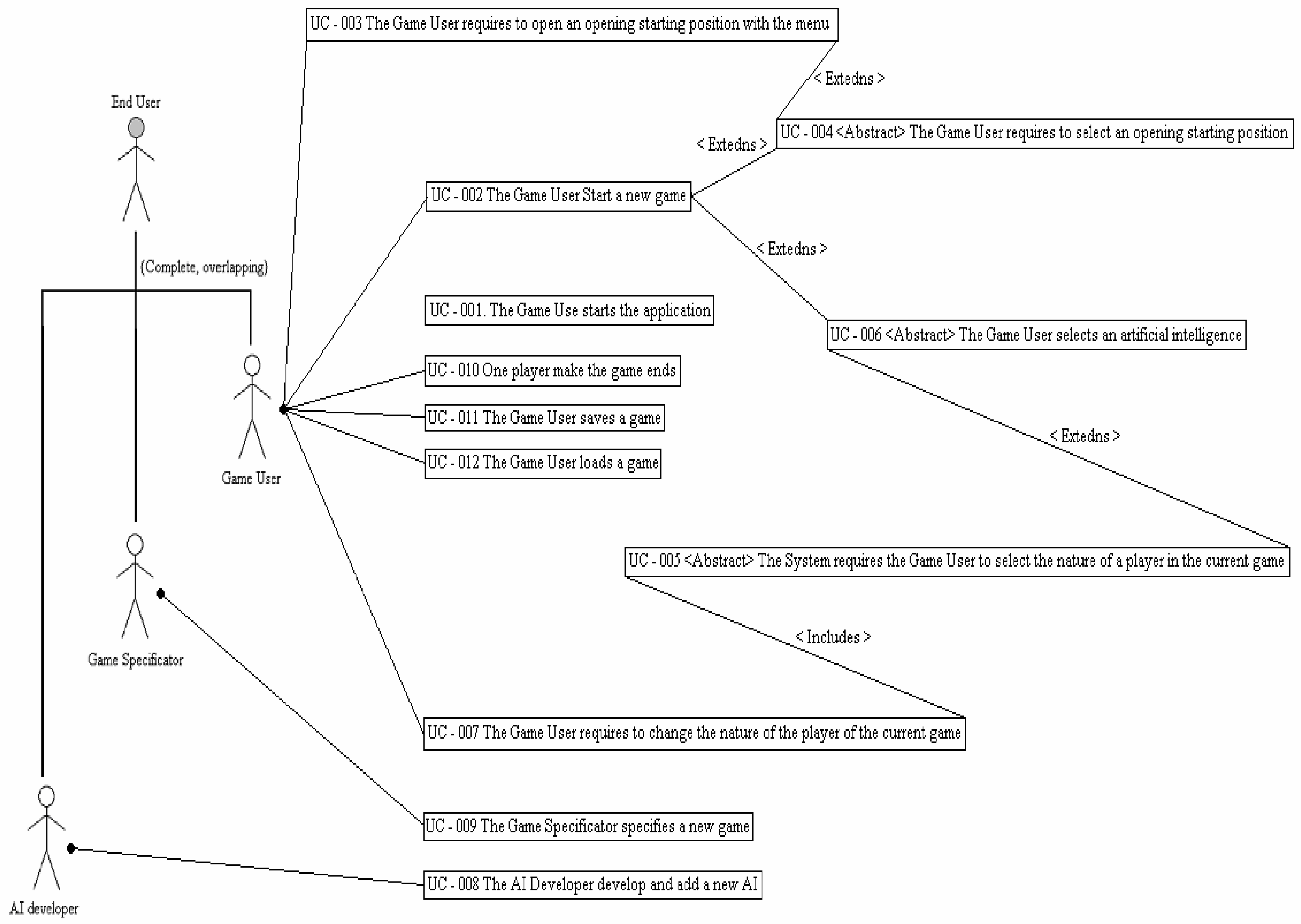
UI-1	All the user interfaces will be designed following common usability guidelines. A specific user interface document will describe the user interface more accurately.
UI-2	The user interface uses one main window for the actual game playing and possibly dialog windows for changing the general settings of the system.
HI-1	The system functions on a computer that has a screen for output and keyboard or mouse or both for input.
SI-1	The system requires a working Java-environment [2] and a graphical user interface supported by the Java Swing-library.
PR-1	The games should be playable on a 450 MHz or higher class computer.
PR-2	The artificial intelligence must operate fast enough. The next move should be calculated within 10 seconds.
SQA-1	The system shall run well on both Linux and Windows and should be playable with the mouse or the keyboard or both.

Appendix C: Stakeholders

Stakeholder	Major Value	Attitudes	Major Interests	Constraints
Researchers	Research value	Seek new research problems	Theoretical problems to be solved	Consistent and scientific layout for the program

Basic user /game player	Entertainment	Relaxing atmosphere	Fun value and interesting game problems to solve	Must run on low-end workstations and be easy to use
Game developer	Platform to test new triangular games	Concerns about the ease of developing games for the platform	Research and easy developing of new games	Documentation needed to understand the methods of developing new games.
Development team	To develop a working program	To learn	Study software project related issues	Easy to use developing environment

Appendix D: Use Case diagram


End User
(Complete, overlapping)
Game User
Game Specificator
AI developer
UC - 003 The Game User requires to open an opening starting position with the menu
< Extedns >
UC - 004 <Abstract> The Game User requires to select an opening starting position
< Extedns >
UC - 002 The Game User Start a new game
< Extedns >
UC - 001. The Game Use starts the application
UC - 006 <Abstract> The Game User selects an artificial intelligence
UC - 010 One player make the game ends
UC - 011 The Game User saves a game
UC - 012 The Game User loads a game
< Extedns >
UC - 005 <Abstract> The System requires the Game User to select the nature of a player in the current game
< Includes >
UC - 007 The Game User requires to change the nature of the player of the current game
UC - 009 The Game Specificator specifies a new game
UC - 008 The AI Developer develop and add a new AI