Concordia University
Engineering and Computer Science

SOEN 6011
Software Engineering Processes

Summer 2016 / AA

ZeroX Game

Assignment 2: Requirement Document

Submitted To: Nicolangelo Piccirilli
Dittimi Tamarafinide

Project Name

- ZeroX Game

Team Name

- Team ZeroX

Team Leader

- Gurvinder Devgun

Team Members

Name	**ID**	***Email Address***
Gurkamal Dhoot	27275978	*g_dhoot@encs.concordia.ca*
Gurvinder Devgun	40012931	*g_devgun@encs.concordia.ca*
Kamaljeet Dhaliwal	27616252	*k_dhali@encs.concordia.ca*
Labdhi Deliwala	27801289	*l_deli@encs.concordia.ca*
Ricardo Cortés	27734107	*r_cort@encs.concordia.ca*
Bhavik Desai	27520239	*b_desa@encs.concordia.ca*
Urvi Desai	27821964	*u_desa@encs.concordia.ca*
Karan Deep Singh	40010405	*ka_ingh@encs.concordia.ca*
Amarpreet Singh	40013505	*s_amarpr@encs.concordia.ca*
Omar Faruk	27291698	*o_faru@encs.concordia.ca*

Document Change Control

Version Number	**Date of Issue**	**Author(s)**	**Brief Description of Change**
V1.0	May 2016	Gurvinder Devgun Karan Deep Singh	Approved Version

Contents

1. Problem Statement

Vision of the project is to make the users realize how to work in teams where you don't know who you are going to work with. Along with this it will familiarize us with the requirement elicitation, gathering and analysis techniques. It will help us to build our collaborative efforts to tackle more complex problems, plan and manage time, refine understanding through discussion and explanation. Furthermore we will be acquainted with different kind of methodology in order to solve a problem in terms of context.

To quickly come over constraints and gel with team mates to make coherent, concise and clear product we will be following real world software engineering processes to mimic the environment that we will be working on when we go over and work in big corporations. Along with this we followed DMAIC (Define, Measure, Analyze, Improve and Control) principles for getting over our problems and to improve our solutions.

2. Background Information

Tic-Tac-Toe is a game that requires a user to a alternatively place X or O's on the 3*3 grid as shown in figure 1 below. Any empty cell in the grid is a candidate for a player to mark his/her symbol. The figure 2 below shows numbered board boxed from 1 till 9.

A winner is declared when a straight line of symbols is created as per below formations:
1) Horizontally: {1, 2, 3} or {4, 5, 6} or {7, 8, 9}
2) Vertically: {1, 4, 7} or {2, 5, 8} or {3, 6, 9}
3) Diagonally: {1, 5, 9} or {3, 5, 7}

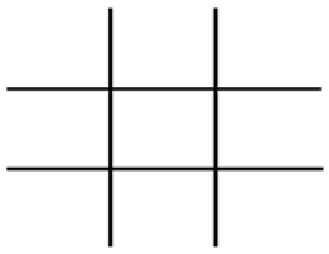
Figure 2.1

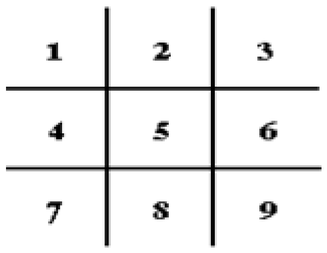

Figure 2.2

The game ends when (1) all the cells are marked, or (2) one of the player wins. These are the basic flows that can also be called the characteristics of a Tic-Tac-Toe game. These characteristics should be present in the final outcome along with the additional ones for enhancing the usability and other functional, non-functional parameters of the application.

We, as a team, are required not just to write a program but also to follow proper project guidelines which mimic a specific software engineering process as defined by the industry standards. The usage of an industry standard software engineering process will help us to manage the project in timely manner and with minimal overhead. With the launch of this application, we think all of the Team members would grasp substantial amount of skills.

3. Functional & Non-Functional Requirements

The following definitions are intended as a guideline to prioritize requirements:

- *Priority 1*: These requirements are "must have".
- *Priority 2*: These requirements are also "must have" however they have lower precedence over Priority 1.
- *Priority 3*: These requirements are "nice to have" as they may include a new functionality that is beneficial to the stakeholders.

3.1. Functional Requirements

A functional requirement describes functionality of system and its components. Functionality consists of set of inputs and its corresponding behaviour and output. The following functional requirements are addressed in design of Tic-Tac-Toe game.

3.1.1. Deliverable 1: Tic-Tac-Toe Board

Deliverable 1 *"A stand-alone Java application that is able to show the board and draw an "X" or an "O" where the user clicks"*				
Req No.	**Description**	**Priority**	**Date Reviewed**	**SME Reviewed / Approved**
1	The application must have a 3*3 board displayed on the screen for the game-play.	1	05/13/2016	Dittimi
2	The application must allow the user to quit the game anytime.	1	05/13/2016	Dittimi
3	The application must allow the user to choose "X" or "O" for display on the game board.	1	05/13/2016	Dittimi
4	The application must allow the user to reset the board.	2	05/13/2016	Dittimi
5	The application must disable an empty grid cell as soon a user clicks it. It must display the symbol associated with the user who clicked in the disabled cell.	1	05/13/2016	Dittimi

3.1.2. Deliverable 2: Porting Tic-Tac-Toe

Deliverable 2: Porting Tic-Tac-Toe *"A Java mobile application that works on Android devices for the full Tic-Tac-Toe game (2 human players)"*				
Req No.	**Description**	**Priority**	**Date Reviewed**	**SME Reviewed / Approved**
1	The Tic-Tac-Toe game must allow two human players to play the game as opponents	1	05/13/2016	Dittimi
2	There must be three possible results – Tie, a Player wins or his opponent wins.	1	05/13/2016	Dittimi
3	A player marks any of the grid squares with his symbol to create a straight line (horizontally, vertically or diagonally) before his opponent in order to win the game or restrict his opponent's move.	1	05/13/2016	Dittimi
4	The application must allow the user to start a new game.	1	05/13/2016	Dittimi
5	The application must display a summary of games won, lost or tied for the user.	2	05/13/2016	Dittimi
6	The application must allow the users to play a series of 3 and series of 5 games.	2	05/13/2016	Dittimi
7	The application must display the player name whose turn is due currently.	2	05/13/2016	Dittimi

3.1.3. Deliverable 3: AI Tic-Tac-Toe

Deliverable 3: AI Tic-Tac-Toe *"A two player computer version of the game against a computer player that uses a heuristic to attempt to beat the human player. Should work on desktop or Android mobile"*				
Req No.	**Description**	**Priority**	**Date Reviewed**	**SME Reviewed/ Approved**
1	The application must allow the user to play against computer as opponent	1	05/13/2016	Dittimi
2	The application must allow the user to play with novice and expert level of computer game play ability.	1	05/13/2016	Dittimi
3	The application must run a background music when a player is playing against computer.	2	05/13/2016	Dittimi
4	The application must allow the user to select either "X" or "O" as his/her symbol during game play.	2	05/13/2016	Dittimi
5	The application must display an e-gift message to the user on win.	2	05/13/2016	Dittimi

3.2. Non-Functional Requirements

Together with functional requirements, non-functional requirements are technical and environmental constraints that the software must meet. For each deliverable, the following non-functional requirements has been address in the design of the Tic-Tac-Toe application.

3.2.1. Deliverable 1: Tic-Tac-Toe Board

"*A stand-alone Java application that is able to show the board and draw an "X" or an "O" where the user clicks"*

- Technology
  - The application must be design and develop based on Java 8.
- Platform Compatibility
  - The application must run on Windows 7, Ubuntu Linux 14, and Mac OS 10 or later versions, on x32 and x64 architectures.
- Documentation
  - Along with the application, a Design, User and Installation manual must be provided. Also, the source code must be documented using Javadoc standards.
- Localization
  - The application must be offered in English.
- Extensibility
  - The design of the application must allow the easy incorporation of the following key future feature:
    - Board of size 5x5, 7x7, 9x9, and 11x11, with the end of game rule of 3 and 5 in line.

3.2.2. Deliverable 2: Porting Tic-Tac-Toe

"A java mobile application that works on Android devices for the full Tic-Tac-Toe game (2 human players)"

- Privacy
  - The application must not request, access, store or distribute any information that identify the User of the mobile device, his/her location and his/her preferences.
- Auditability
  - The application must keep a detailed log of its execution and user interaction, so it can be used when analysis its working during maintenance activities. The log must not exceed 25MB of storage.
- Compatibility
  - The application must be accepted in an Android Application Store (e.g. Google Play). In other words it must be ready for the publishing process (i.e. follow the development standards, have

the promotion material, understand the policies and agreements, have verify the quality of the application, etc.)

- Portability
    - The application must adapt its user interface to support mobile devices with display aspect ratio of 4:3 (e.g. 2048x1536), 16:9 (e.g.1920×1080) and 16:10 (e.g. 1920x1200)
- Extensibility
    - The design of the application must allow the easy incorporation of the following key future feature:
        - iOs support, for versions 8 and later.

3.2.3. Deliverable 3: AI Tic-Tac-Toe

"A two player computer version of the game against a computer player that uses a heuristic to attempt to beat the human player. Should work on desktop or Android mobile."

- Resource
    - In any moment, the application must never use more than 10% of CPU, and it must never use more than 256MB of RAM. Also, the application must never use, in any circumstance, mobile data of the device.
- Response Time
    - When playing against the device, the application must present a result within 3 seconds. When performing other actions (e.g. accessing menu options, selecting configurations), the application must respond within 1 second.
- Licensing
    - The application both, desktop and mobile, must be design and develop using Open Source licensing principles (e.g. GPL). Also, any copyright must be properly acknowledge in its documentation or application information.
- Interoperability
    - The application must be designed and developed so the User Interface must be decoupled from the Tic-Tac-Toe Algorithms, so the change of the UI or the implementation of new game algorithms can be easily done (e.g. Chess Engines).
- Extensibility
    - The design of the application must allow the easy incorporation of the following key future feature:
        - Web version, multiplayer game, multiple device.

4. Use Case for Deliverable - 1

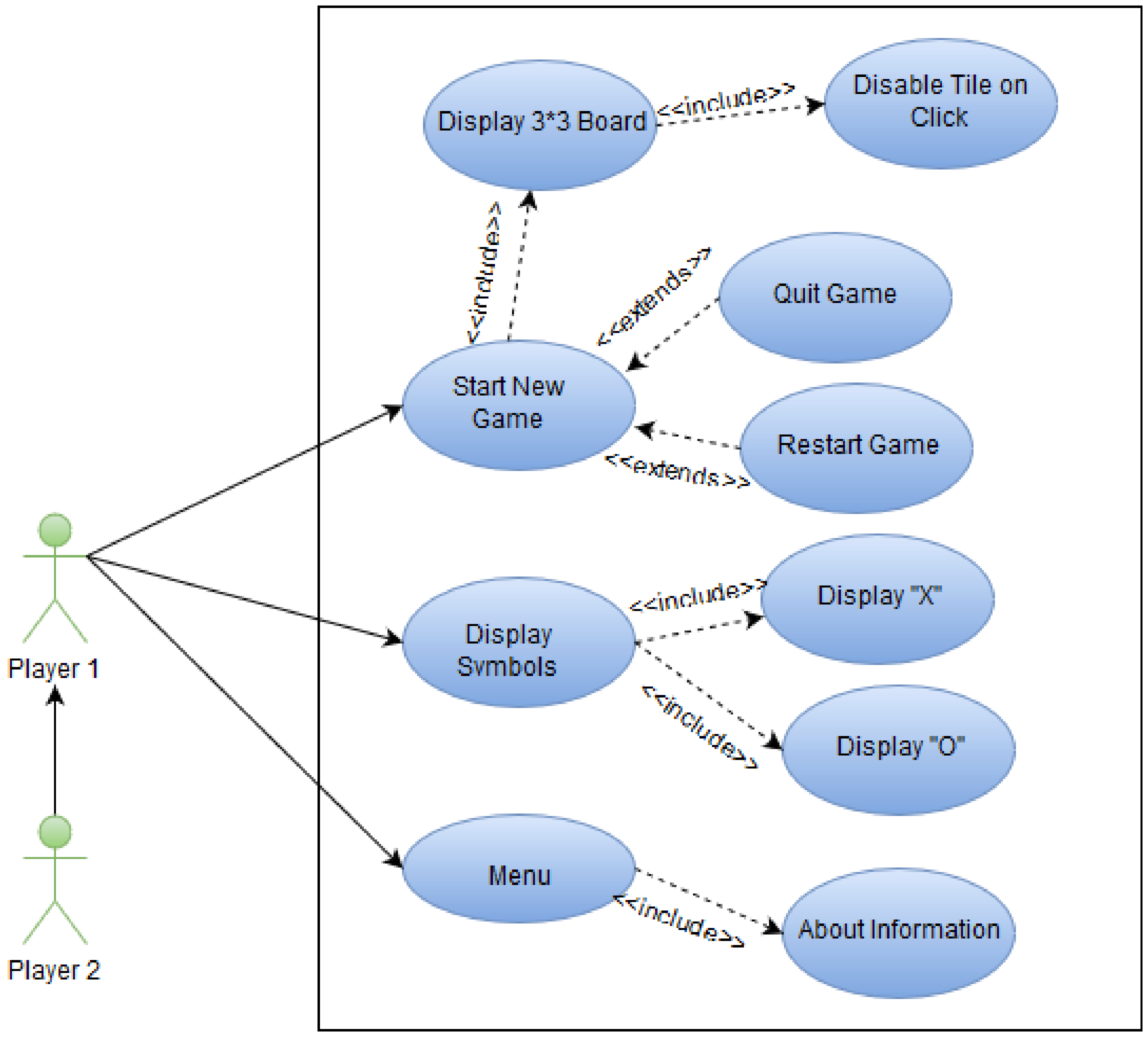

Figure 4.1

As depicted in Figure 4.1, following Use-Case are created:

- UC_1_1: Start New Game
- UC_1_2: Quit Game
- UC_1_3: Restart Game
- UC_1_4: Display Symbols
- UC_1_5: Display 3*3 board
- UC_1_6: Disable Tile on Click
- UC_1_7: Menu

4.1. Use case UC_1_1: Start new game

This use case describes how the user starts and navigates through the application.

Level: User goal
Primary Actors: Human Player (Player 1 & Player 2)
Supporting Actors: Tic-Tac-Toe Windows application

Stakeholders and interests
- Human players want to start and play the Tic-Tac-Toe game
- Students as developers want to develop an interesting game.

Pre-Conditions
- Installation of the game
- Compatible system to run the game
- User starts the application.

Post-Condition
- User quits the game.

Success Condition
- Player successfully navigates through the application menu and is able to click on desired cell of the 3*3 grid.

Failure Condition
- The grid does not records the user input and the application hangs.

Main Success scenario:
1) The User clicks on start game.
2) The system starts the game and displays 3*3 grid to the user.
3) The user clicks on grid cell.
4) The system displays “X” symbol on the first click.
5) The second user clicks on an empty grid cell.
6) The system displays “O” symbol for the second user.
7) The User clicks on grid cell that is not empty.
8) The system does not allow the user input on the filled grid cell.
9) The user clicks on the quit game to exit the game.

Alternate flow:
7a) The System records user click on an already filled grid cell.

Sub-Flow: The flow can move from UC_1_1 i.e. Start New Game to UC_1_2 to quit game and to UC_1_3 to Restart Game.

4.2. UC_1_2: Quit Game

Level: Sub-function
Primary Actors: Human Player (Player 1 & Player 2)
Pre-Condition: The user is playing the game.

Main Flow: The user while playing the game decides to quit the application. The player clicks on the Quit button to exit the application. The system records the user input and closes the application.

4.3. UC_1_3: Reset Game

Level: Sub-function
Primary Actors: Human Player (Player 1 & Player 2)
Pre-Condition: The user is playing the game.

Main Flow: The user while playing the game decides to reset the application. The player clicks on the Reset button to reset the board. The system records the user input and refreshes the board.

4.4. UC_1_4: Display Symbols

Level: User Goal
Primary Actors: Human Player (Player 1 & Player 2)
Pre-Conditions: The User has started the game.

Main Flow: The user has started the game and clicks on the board to record symbol. The system receives the input and displays "X" for the first click. The second user then clicks the board and the system displays "O" for the next click.

4.5. UC_1_5: Display 3*3 Board

Level: User Goal
Primary Actors: Human Player (Player 1 & Player 2)
Pre-Condition: The User has started the game.

Main Flow: The user clicks on start new game. The system receives the user input and displays the 3*3 grid for the Tic-Tac-Toe game.

Sub-Flow: The flow moves from UC_1_5 to UC_1_6 which disables the empty Tiles on click.

4.6. UC_1_6: Disable Grid Cells

Level: Sub-function
Primary Actors: Human Player (Player 1 & Player 2)
Pre-Conditions: The User has started the game.

Main Flow: The User clicks on tile to record his/her symbol. The system records the inputs and disables the tile.

4.7. UC_1_7: Menu

Level: User Goal
Primary Actors: Human Player (Player 1 & Player 2)
Pre-Condition: The User has started the game.

Main Flow: The user wants to see Menu options and clicks on the Menu button. The system records the input and displays Menu options.

5. Use Case for Deliverable - 2

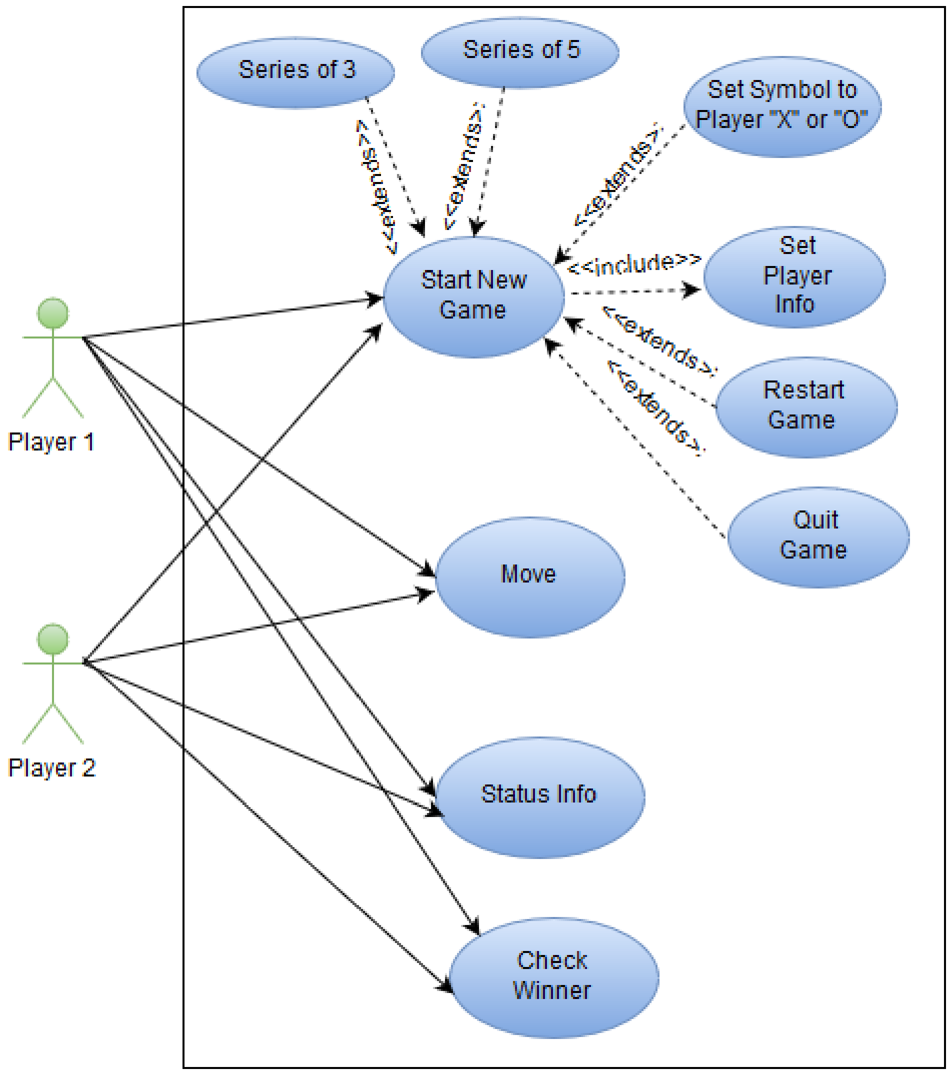

Figure 5.1

As depicted in Figure 5.1, following Use-Case are created:

- UC_2_1: Move
- UC_2_2: Set Player Info
- UC_2_3: Set Symbol to Player X or O
- UC_2_4: Status Info
- UC_2_5: Check Winner
- UC_2_6: Series of 3
- UC_2_7: Series of 5

5.1. UC_2_1: Move in a Multiplayer Game

This use case describes the interaction scenario for multiplayer game in an android device..

Level: User Goal

Primary Actors: Human Player (Player 1 & Player 2)

Supporting Actors: Smart Phone (Android technology supported)

Stakeholders and Interests

- Human Players want to want to play the game against each other.
- Professor as a stakeholder requires an android Tic-Tac-Toe game with multiplayer capabilities.
- Students as developers want to develop Tic-Tac-Toe game.

Pre-Conditions

- Players have android devices.
- Android application is installed on the player’s device.
- Player has launched the application.

Post Condition

- Players exit the game.

Success Condition

- Players successfully play the game and can see the winner.

Failure Condition

- Game crash when playing or when starting to play

Main Success Scenario:

1. Click start button on displayed screen to start the game.
2. Set the player name
3. Choose the symbol for playing (X or O)
4. Player 1 makes the move.
5. Marked grid cell as blocked with Player 1 symbol.
6. Player 2 makes the move
7. Repeat the step 4 to 6 until the game finish.
8. Show the status of game at finish.

Extensions:

4a. Show the player name on top of screen bar.

6a. Show the player name on top of screen bar.

Sub-Flow: At the start of the game the user can set symbol to Player X or O.

5.2. UC_2_2: Set Player Info

Level: Sub-function.
Primary Actors: Human Player (Player 1 & Player 2)
Pre-Condition: The user has opened the Tic-Tac-Toe android application on device.

Main Flow: The User i.e. the player clicks on the Set Player Info. The system acknowledges the inputs. The user then stores his information and the system confirms.

5.3. UC_2_3: Set Symbol to Player X or O

Level: Sub-function
Primary Actors: Human Player (Player 1 & Player 2)
Pre-conditions: Player has opened the Tic-Tac-Toe application on android device.

Main Flow: The user selects the symbol X to represent his/her turn. The system acknowledges the user request and assigns the requested symbol.

5.4. UC_2_4: Status Info

Level: Sub-function
Primary Actors: Human Player (Player 1 & Player 2)
Pre-Conditions: Both the players have finished playing at least 1 game.

Main Flow: The Player 1 starts a new game with Player 2. The system records this event and updates the scorecard with the status from the previous games.

5.5. UC_2_5: Check Winner

Level: Sub-function
Primary Actors: Human Player (Player 1 & Player 2)
Pre-Conditions: The game is in progress between two human players.

Main Flow: The User makes a move and the system acknowledges. The system checks for winner on each move.

5.6. UC_2_6: Series of 3

Level: Sub-function

Primary Actors: Human Player (Player 1 & Player 2)

Pre-Conditions: User is on home screen of the application

Main Flow: The User selects series of 3. The system records the input. The use then clicks on start game. The system records the input and prepares the game for a series of 3 game.

5.7. UC_2_7: Series of 5

Level: Sub-function

Primary Actors: Human Player (Player 1 & Player 2)

Pre-Conditions: User is on home screen of the application

Main Flow: The User selects series of 5. The system records the input. The use then clicks on start game. The system records the input and prepares the game for a series of 5 game.

6. Use Case for Deliverable - 3

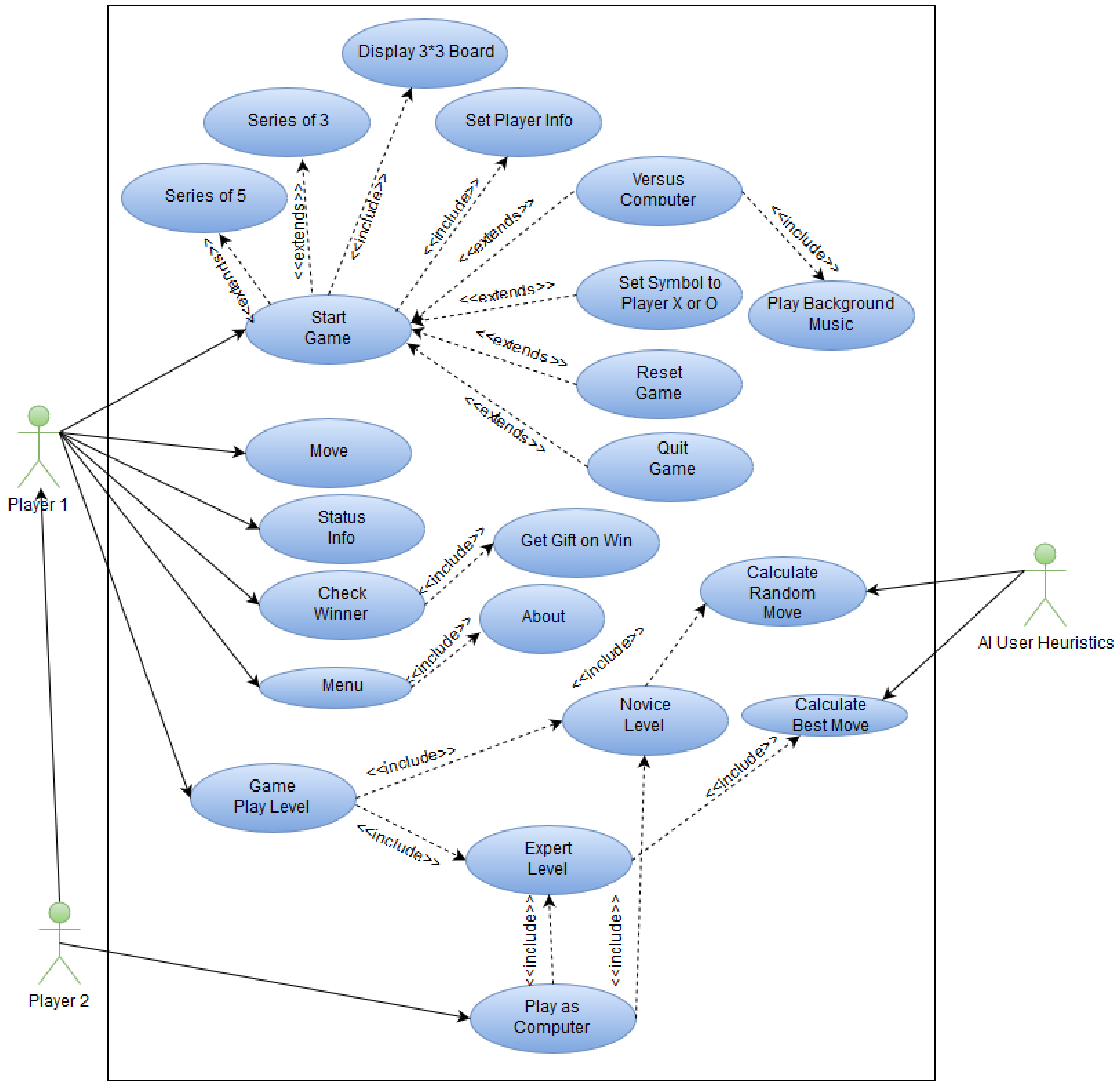

Figure 6.1

As depicted in Figure 6.1, following Use-Case are created:

- UC_3_1: Versus Computer
- UC_3_2: Play Background Music
- UC_3_3: Get Gift on win
- UC_3_4: Game Play level
- UC_3_5: Novice Level
- UC_3_6: Expert Level

- UC_3_7: Calculate Random Move
- UC_3_8: Calculate Best Move
- UC_3_9: Play as computer.

6.1. UC_3_1: Play versus Computer

This use case describes the interaction scenario when a human player opts to play against a computer player. When the players want to play this game they have to start the game.

Level: User Goal
Primary Actors: Human Player (Player 1 & Player 2)
Supporting Actors: AI User Heuristics

Stakeholders and Interests

- Human Players want to want to play the game.
- Professor requires a Tic-Tac-Toe game where a human player can compete against Computer player.
- Students as developers want to develop Tic-Tac-Toe game.

Pre-Conditions

- Player needs an android device to run this application.
- Player has to install this game to play.
- Click in application icon to start the application

Post Condition

- Players can quit the game.

Success Condition:

- A player successfully plays and exits the game.

Failure Condition:

- Game crash when playing or when starting to play.

Main Success Scenario

1. Click start button on displayed screen to start the game.
2. Set the player name
3. Choose the symbol for playing (X or O)
4. Select to play against computer.
5. Player 1 makes the move
6. Marked grid cell as blocked with Player 1’s symbol.
7. Computer player makes the next move.
8. Repeat the step 5 to 7 until the game finish.
9. Show the status of game at finish.

Extensions:

5a. Quit right after starting the game without playing.

7a. User refreshes the board.

6.2. UC_3_2: Play Background Music

Level: Sub-function

Primary Actor

- Human Player
- Computer Player

Pre-Conditions:

- User has installed the applications on his device.
- The User is currently on the home page.

Main Flow: The user selects to play against Computer and starts a game. The system records this input and starts music in the background.

6.3. UC_3_3: Get Gift on Win

Level: Sub-function

Primary Actor

- Human Player
- Computer Player

Pre-Conditions:

- User has installed the applications on his device.
- The User is playing a game with computer user.

Main Flow: The user wins the game. The system records the transaction and displays a congratulation message along with a image of a gift pack.

6.4. UC_3_4: Novice Game Play level

Level: User Goal

Primary Actor

- Human Player
- Computer Player

Secondary Actor: AI User Heuristics.

Pre-Conditions:

- User has installed the applications on his device.
- User is on the home page of the application.

Main Flow: The user selects the game play level to Novice and starts the game. The system records the transaction and set up the AI User heuristic to Novice Level.

6.5. UC_3_5: Expert Game Play Level

Level: User Goal
Primary Actor

- Human Player
- Computer Player

Secondary Actor: AI User Heuristics.
Pre-conditions:

- User has installed the applications on his device.
- User is on the home page of the application.

Main Flow: The user selects the game play level to Expert and starts the game. The system records the transaction and set up the AI User heuristic to Expert Level.

6.6. UC_3_6: Calculate Random Move

Level: Sub-function
Primary Actor

- Human Player
- Computer Player

Secondary Actor: AI User Heuristics.
Pre-conditions:

- User has installed the applications on his device.
- User is playing game with Computer at Novice Level.

Main Flow: The user selects his move and blocks a grid cell where he/she makes the move. The Computer makes a random move without checking any heuristics.

6.7. UC_3_7: Calculate Best Move

Level: Sub-function
Primary Actor

- Human Player
- Computer Player

Secondary Actor: AI User Heuristics.
Pre-conditions:

- User has installed the applications on his device.
- User is playing game with Computer at Expert Level.

Main Flow: The user selects his move and blocks a grid cell where he/she makes the move. The Computer calculates the best move based on heuristics and makes the move.

6.8. UC_3_9: Play as Computer.

Level: User Goal
Primary Actor

- Human Player
- Computer Player

Secondary Actor: AI User Heuristics.
Pre-conditions:

- User has installed the applications on his device.
- User on the home page of the application.

Main Flow: The user selects Player 2 as Computer. The system records the transaction. The user selects the game play level to be novice and clicks on start game. The System records user input and starts the game.