Software Requirements Specification

For

Tic-Tac-Toe Game Application

Prepared by: Phaneendra Reddy C

Contents

Revisions

1 Introduction

1.1 Document Purpose

1.2 Product Scope

2 Overall Description

2.1 Product Perspective

2.2 Product Functionality

2.3 Users and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

3 Specific Requirements

3.1 External Interface Requirements

3.2 Functional Requirements

3.3 Behaviour Requirements

4 Other Non-functional Requirements

4.1 Performance Requirements

4.2 Safety and Security Requirements

4.3 Software Quality Attributes

5 Other Requirements.

Introduction:

Tic-tac-toe, is a pencil-and-paper game for two players, O and X, who take turns marking the spaces in a 3×3 grid, usually X going first. The player who succeeds in placing three respective marks in a horizontal, vertical, or diagonal row wins the game or In Tic-Tac-Toe a game can end with a player winning (getting 3 of his pieces in row) or end in a draw (no spaces left on the board with no-one winning).

1.1 Document purpose:

The Tic-Tac-Toe Game Application is changed from a pencil-paper game to a computer game where 2 players sit at a computer and play the same game.

1.2 Product Scope:

The Tic-Tac-Toe game is changed into a 2 player game on a computer-one plays with X’s and the other as O’s. Both these players play on the client’s side. The server program evaluates the game and declares the winners or if the game is a draw.

This game is implemented in client server manner. 2 players will be playing at the client side-one playing as ‘X’ and the other playing as ‘O’. The server will check and validate whether or not the game is a win or a draw. If it is a win, the server declares which player is the winner. The program is implemented in C#.

1.3 Intended Users :

The document is intended to be used by the programmers and document writers.

2.OVERALL DESCRIPTION :

2.1 Product perspective :

This product is eventually developed for Non-Technical people / those Technical people who are spoiling their health by getting addicted to smoking.

the minimax algorithm returns the best possible outcome for the move so an user always thinks of new algorithm to win the game and eventually gets addicted to this game.

2.2 Product Functionality:

1. The game has to be user friendly.
2. Instructions must be provided, which help the user throughout the game. This is useful for new users.
3. Hints must be provided when the player requests for it.
4. Game sounds should be heard at the correct time.

5. The player can either win or draw a game.So scores should be given accordingly.

2.3 Users and characteristics:

The 2 players who play the game are the most important users of the system. All the above functionalities are relevant to these players.

2.4 operating Environment:

The game should be able to work on all platforms. It must coexist peacefully with the music system which provides game sounds. It should also support a help system that provides instructions or hints to players

2.5 Design and implementation constraints :

* This game can be played only by 2 players on a single system.
* Once a game is started, the user can only play the game or quit the game. He cannot open a new game.
* If the game is a win or a lose, the server should immediately display the winner, that is the server should be fast enough to display the winner without allowing the other player to make a move.

2.6 User documentation:

The game application should be able to provide instructions for new players and provide hints, on demand, to the player who has to make a move.

2.7 Assumptions and dependencies :

* There are 2 players playing the game on a single system.
* The sound system on the system is working properly to enable game sounds.
* Once the game starts, it is not interrupted in the middle.

3,SPECIFIC REQUIREMENTS :

3.1 External Interface Requirements

3.1.1 User Interfaces:

The user interface for the Tic-Tac-Toe game application is the game board. It is from this board that a user plays the game. On winning a game, it should display a dialog box showing who the winner is.

3.1.2 Hardware Interfaces :

Music system is interfaced to our game application.

3.1.3 Software Interfaces:

An interface game board containing blocks for x and o’s for a player to play the game is provided.

3.2 Functional Requirements:

1. **Displaying a 3x3 grid:** This grid is required to play the Tic-Tac-Toe game. It is on this grid that the two players take turns marking the spaces as X and O.
2. **Alternating between X and O Players:** This game requires two players, one playing as X and one playing as O (with X usually starting). To ensure that a player doesn’t play twice, switching between them is required.
3. **Determining the Winner of the Game:** Once a player wins the game, the message should be displayed which player won. If no one won the game, then it is a draw.
4. **Displaying Score Details:** There must be 3 columns for maintaining the scores. One for each player and one for the games which were a draw.
5. **Providing Option to Play Again:** The game must provide an option to either play again or to exit the game.

3.3 Behaviour Requirements:

3.3.1 Use case views :

Actors: roles that users (or systems) play

– Actors carry out use cases

– A single user could play several roles

\_multiple users could play the same role.

Scenarios for the ‘Make Move’ use case:

– A Game Player places a symbol on an open

square on the Game Board

– A Game Player places a symbol on an occupied

square on the Game Board

• Original symbol continues to occupy square

• Allow Game Player to select another square.

4.OTHER NON-FUNCTIONAL REQUIREMENTS:

4.1 Performance Requirements:

1. **Responsiveness:** The game application must respond quickly to the players and changing environment.

4.2.Software Quality Attributes:

1. **Responsiveness:** The game application must respond quickly to the players and changing environment.
2. **Usability:** The game application must be easy to use so that even the new players can adapt easily.
3. **Reliability:** Ability to retrieve the scores even after the game application crashes.
4. **Portability:** The application must be portable so that it can run on any platform.
5. **Robustness:** The application must be robust to allow the scope of making mistakes by the players.

5.Other Requirements:

1. **Providing Game Sounds:** The game must also provide background sounds. The user also has the facility to mute the sounds whenever he/she pleases.
2. **Providing Instructions to play the Game:** For players who are new to the game, instructions are given, so that they too can play.
3. **Providing Hints:** Hints are to be given whenever the player requests hints.