SE 3XA3: Software Requirements Specification
KingMe

Team 9, KingMe
Ardhendu Barge 400066133
Dylan Smith 001314410
Thaneegan Chandrasekara 400022748

February 12th, 2021

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
2021-02-07	0	Completed section 1
2021-02-11	0	Dylan added FR
2021-02-12 Feb	0	Ardhendu and Thaneegan finished the document
2021-04-10	1	Ardhendu reviewed FR and removed FR5
2021-04-10	1	Dylan removed NFR3
2021-04-11	1	Dylan added numbering to the NFRs
2021-04-11	1	Thaneegan removed NFR17
2021-04-12	1	Thaneegan added FR17
2021-04-12	1	Ardhendu removed NFR2

1 Project Drivers

1.1 The Purpose of the Project

The purpose of this project is to re-implement the checkers game with a Graphical User Interface and adding any missing parts such as proper game instructions fix game logic and add more options for the player. The Artificial Intelligence algorithm for computer play will be optimized for better gameplay and user experience. In addition, missing king piece will be added to make the game regulation size. The game will be reorganized to follow formal software engineering quality attributes such as usability, learnability, maintainability, portability, etc.

1.2 The Stakeholders

1.2.1 The Client

The client for the checkers game are the teaching staff, specifically the professor and the teaching assistants for the Software 3XA3 course. Throughout the project life-cycle, the client will be involved in providing feedback and reviewing the product implemented.

1.2.2 The Customers

The customers for the checkers game include anyone who is interested in learning how to play, or anyone looking to improve their existing skills or just enjoy the game.

1.2.3 Other Stakeholders

In addition to the client and customers specified above, another group of stakeholders include chess organizations and developers who will work on this open-source project in the future. Checkers organizations can range from school clubs to professional clubs that are looking to transition in-person play to online. Future developers are an indirect stakeholder group, as they will be interested in developing a well-documented, logical project.

1.3 Mandated Constraints

- The checkers game will be based on the open source project by codeofcarson.
- The project must be completed by April 12 2021.
- The project must be able to run on the McMaster Mills server.
- The budget for the project is $0, which will affect the quality of product being developed.

1.4 Naming Conventions and Terminology

- GUI: Graphical User Interface
- AI: Artificial Intelligence
- UI: User Interface

1.5 Relevant Facts and Assumptions

1.5.1 Facts

- The existing open-source project does not have certain checkers rules implemented.
- The existing open-source project is a smaller version of checkers that does not have all the pieces and is played on a smaller board.
- The existing open-source project has 388 lines of code.

1.5.2 Assumptions

- The user has a device to play the chess game on.
- The user's device has Python downloaded in order to run the application.
- The user is capable of reading English and following instructions written in English.
- User is able to use computer and internet

2 Functional Requirements

2.1 The Scope of the Work and the Product

The software product being built is a command line version of the game of checkers. The game will be improved from its previous version by updating it to include all 24 checkers pieces as opposed to the 12 pieces currently implemented. The checkers game will be implemented to follow all the rules of checkers, a tutorial will be made available to users to help beginners learn how the rules. A Graphical User Interface will also be added to improve the playability of the game. If time permits alternate versions of checkers will be added to make the app appeal to more people.

2.1.1 The Context of the Work

The context of the checkers app is going to be a command line application. It will be made using the python programming language. External libraries such as pygame will be used to help developers with the construction of the Graphical User Interface.

2.1.2 Work Partitioning

Table 2: **Work Partitioning**

Event Name	**Input**	**Output**	**Description**
Start Game	User Input	State creation, Graphic creation, Display change	User starts game and system outputs the initial state of the board
View Tutorial	User Input	Graphic creation, Display change	User clicks on tutorial and the system outputs the rules of checkers
Make a move	User Input	State creation, Graphic creation, Display change, AI	When a user makes a move the system will update and display the board and then the AI will make a move
Leave game	User Input	Graphic creation, Display change	User chooses to leave game and the system returns to the home screen

2.1.3 Individual Product Use Cases

Case UC-1: Play ckeckers
Related Requirements: FR1-FR9
Initiating Actor: The user
Actor's Goal: Play a game of checkers.
Participating Actors: None
Preconditions: No game is in progress, system is displaying the home screen.
Postconditions: Either the win or loss state is achieved
Flow of Events for Main Success Scenario:
→ 1. The initiating user chooses to start a game.
← 2. System prompts user to choose their opponent (either the computer or another player).
→ 3. The user chooses their opponent.
← 3.1 If the user chooses to play the computer the system prompts them to choose the colour of their pieces.
→ 3.2 User chooses their colour (if they chose to play the computer).
← 4. System displays the state of the board.
↔ 5. The player with the black pieces moves first.
← 6. The system updates the state of the board.
↔ 7. The player with the red pieces moves next.
← 8. The system updates the state of the board.
← 9. Players alternate turns until a win/loss state is achieved.
Flow of Events for Alternative Scenarios:
Leave game:
← 1. The user chooses to leave the current game.
→ 2. The system returns to home screen.
Invalid move:
← 1. The user attempts to make an invalid move.
→ 2. The system displays and error message.
→ 3. The system returns the state of the board to what it was before the invalid move was attempted.

2.2 Functional Requirements

Rules of the checkers game were referred to via [1]

FR1. **Description**: The system shall have a home screen that provides op-

tion to user to start a game or view the checkers tutorial.
Rationale: System shall allow the user to start a game or view the tutorial whenever they want.
Fit Criterion: The system remains on the home screen until the user initiates an event.
Originator: Dylan Smith added February 11, 2021.

FR2. **Description**: The system will allow the user to choose their colour.
Rationale: To give the user the option to play first or second.
Fit Criterion: When the user selects the "1-Player" mode the system waits for the user to choose their colour.
Originator: Dylan Smith added February 11, 2021.

FR3. **Description**: The system will only allow the User/AI to make valid moves.
Rationale: To abide by the rules of checkers.
Fit Criterion: The User/AI can only choose from a set of valid squares.
Originator: Dylan Smith added February 11, 2021.

FR4. **Description**: The system will show all valid moves for the piece the user selects.
Rationale: This will help the user find valid moves if they are just learning how to play.
Fit Criterion: When the user selects a piece to move the system displays all valid moves.
Originator: Dylan Smith added February 11, 2021.

FR5. ~~**Description**: The system shall display an error message if an invalid move is attempted. **Rationale**: The user needs to know that the move they attempted is invalid, that it won't be accepted, and that it remains their turn.~~
~~**Fit Criterion**: An error message is displayed when an invalid move is made.~~
~~**Originator**: Dylan Smith added February 11, 2021.~~

FR6. **Description**: The system shall display the state of the board after each move.

Rationale: The user will need a way to keep track of the moves that have been made.
Fit Criterion: When a move is made the system will display the move on the board.
Originator: Dylan Smith added February 11, 2021.

FR7. **Description**: The system shall provide users with the option to leave the current game.
Rationale: The user may have chosen "1-Player" and may wish to change to "2-Player mode without having to play the game to completion.
Fit Criterion: The system provides the user with the option to leave the game and return to the home screen.
Originator: Dylan Smith added February 11, 2021.

FR8. **Description**: The user shall win once it captures all the pieces of the opponent.
Rationale: Signifies the win state for user.
Fit Criterion: All of the opponents pieces have been removed from the board.
Originator: Ardhendu added February 12, 2021.

FR9. **Description**: The user shall lose once opponent captures all the pieces of the user.
Rationale: Signifies the loss state for user.
Fit Criterion: All of the users pieces have been removed from the board.
Originator: Ardhendu added February 12, 2021.

FR10. **Description**: The system will display a win/loss screen when a win/loss state is achieved.
Rationale: To signify the end of the game.
Fit Criterion: The system displays the win/loss screen when a final state is reached.
Originator: Dylan Smith added February 11, 2021.

FR11. **Description**: The system must remove a user/opponents piece when it is jumped over by another piece.
Rationale: To display the correct state of the game.

Fit Criterion: The piece that was jumped over will no longer appear on the board.
Originator: Thaneegan added February 12, 2021.

FR12. **Description**: User can only control their pieces during their turn.
Rationale: To prevent user from controlling opponents pieces.
Fit Criterion: The user is not active during the opponent's turn.
Originator: Thaneegan added February 12, 2021.

FR13. **Description**: ~~Black~~ Red side shall be the first one to move a piece.
Rationale: To abide by the game rules.
Fit Criterion: Only ~~black~~ Red side will be able to choose pieces on the first turn.
Originator: Ardhendu added February 12, 2021

FR14. **Description**: Any piece on the board shall not move backwards except king piece
Rationale: To abide by the game rules.
Fit Criterion: When the user selects a non-king piece to move, the system only allows forward moves.
Originator: Ardhendu added February 12, 2021

FR15. **Description**: Pawn piece shall be promoted to king piece if it makes to opposite side of the board
Rationale: To abide by the game rules.
Fit Criterion: Pawn piece replaced by king piece on board.
Originator: Thaneegan added February 12, 2021.

FR16. **Description**: A piece selected by the user shall be highlighted.
Rationale: To provide the user with a visual that a piece has been selected.
Fit Criterion: The desired piece has been highlighted.
Originator: Thaneegan added February 12, 2021.

FR17. **Description**: User will have the option to play against computer or against another user.
Rationale: To provide the user with different game modes.
Fit Criterion: The menu contains two game modes to choose from.
Originator: Thaneegan added April 12, 2021.

3 Non-functional Requirements

3.1 Look and Feel Requirements

- NFR1 The game shall be visually appealing to 90% of users or more.
- ~~NFR2 The game display shall be able to scale in size as the user desires.~~

3.2 Usability and Humanity Requirements

- ~~NFR3 The game will be usable by users with color blindness.~~
- NFR4 The game will be usable by users who can operate a computer.
- NFR5 ~~The game shall have a tutorial for new users to learn the rules.~~ The tutorial will be helpful to at least 80 percent of users.

3.3 Performance Requirements

- NFR6 The game must move a user's piece within 2 seconds of the user requesting that move.
- NFR7 The game must move a computer's piece within 3 seconds of the user completing their turn.
- NFR8 The game must be started and displayed within 5 seconds of the user requesting to play a game.
- NFR9 The game shall run at a frame rate of 60 frames per second or above.

3.4 Operational and Environmental Requirements

- ~~NFR10 The user must have python installed on their device.~~
- NFR11 The game shall run on the user's device without the need for internet.

3.5 Maintainability and Support Requirements

- NFR12 The game shall be well documented.
- NFR13 The game must be modularized into different components.
- NFR14 The code must follow same coding style throughout.

3.6 Security Requirements

- NFR15 The game must not request the user for personal information.
- NFR16 The game must not request the user to download nor install anything.

3.7 Cultural Requirements

- N/A

3.8 Legal Requirements

- ~~NFR17 The game must not use any symbols that are owned by other creators.~~

3.9 Health and Safety Requirements

- N/A

4 Project Issues

4.1 Open Issues

There are several existing issues with the original implementation that significantly impact the ability to play and enjoy the game:

- The lack of a GUI results in a poor appearance to the user. Instead of having a user-friendly GUI, the original implementation forces the user to use the command-line interface to make moves. This affects not only appearance but also performance as well.

- The original implementation is not intuitive as the game does not have sufficient instructions nor rules to help the user understand the game.
- The game does not run on devices with python versions 3.x.x, and instead only run on devices with older versions of python installed such as the python 2.x.x versions. This impacts the usability of users wishing to play the game and acts as a barrier to entry for those who do not wish to install an older version of python.
- The game does not use the standard checkers board or the standard number of pieces. This limited version of checkers will limit interest from users.

4.2 Off-the-Shelf Solutions

Checkers is a very popular game and thus there exists multiple online or downloadable versions. Each off-the-shelf solutions varies in detail, but the core revolves the same rules of checkers. The following are a few examples of checkers games online:

- https://cardgames.io/checkers/
- https://www.gametop.com/download-free-games/checkers/
- https://www.247checkers.com/
- https://www.malavida.com/en/soft/checkers-deluxe/gref
- https://gametable.org/games/checkers/

There are also solutions that are available on GitHub pages or PyPi to download and play. An example of this can be seen below:

- https://pypi.org/project/imparaai-checkers/

4.3 New Problems

As describe in section 4.1 (open issues), there are several issues with the original implementation. As these issues are worked on and fixed, new issues may arise. Although we plan on implementing the best solution possible, there may be a few hypothetical new problems that occur:

- The transition from the original implementation's command-line interface UI to a GUI instead may result in performance issues. Although we aim to have quick performance, as mentioned in our NFRs, the use of a GUI will increase the response time required and we will have to implement an efficient algorithm to prevent this problem from arising.

- Another problem that may arise after introducing a GUI is not displaying the GUI properly due to various user screen sizes. The GUI must be flexible and stretchable in both X and Y directions such that the resolution of the images/components on screen do not look worse. Although we will implement the solution to avoid this problem, time may be a constraint in testing for this.

4.4 Tasks

The tasks for this project are specified by the stakeholder Dr.Bokhari. These tasks, along with the deadline dates, can be found on our Gantt tasks chart available here: Project Tasks. The Gantt task chart can also be viewed down below:

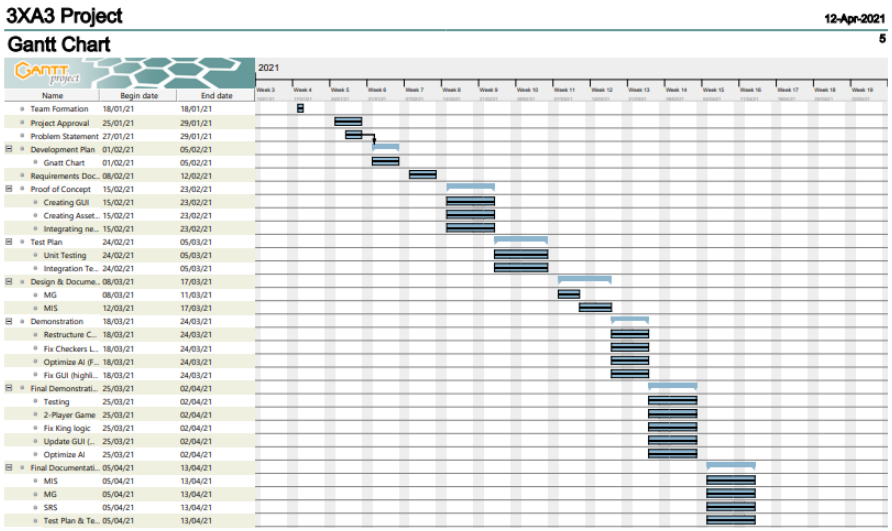

Figure 1: Project Gantt Chart

4.5 Migration to the New Product

The current open-source project supports Python version 2.x.x but does not support Python 3.x.x. Thus we will be migrating the existing code to Python 3 to resolve this.

4.6 Risks

Risk	Probability of Occurring	Level of impact
Testing does not discover all errors/bugs that were made during development before production release.	High	Medium
The GUI developed does not adjust accordingly when the user stretches the GUI in the X or Y directions.	Medium	Medium
Response time (as specified in NFRs) is not met successfully.	Medium	Low
Transition from supporting Python 2 to Python 3 is not completed.	Low	Low

4.7 Costs

There is no monetary costs involved in this version of the project. The features being added to the existing open-source project do not require any money nor do any of the team-members require compensation. However in terms of cost of effort/time, the project Gantt chart illustrates the distribution of work among the team resources. Please refer to our Gantt resource chart for this project here: Project Schedule.

4.8 User Documentation and Training

User documentation and training will come in the form of two methods:

- Walk-through tutorial: The walk-through tutorial will be an optional explanation on how to play the game and where various buttons are on the UI.
- Rules page: There will also be an option for the user to click to view/read the rules of checkers on their own.

4.9 Waiting Room

There are a few requirements that are on the waiting list to be included in future versions:

- Difficulty of Opponent: A feature to allow the user to select difficulty of opponent (computer). There would be a range of difficulty levels to accommodate for different users with different skill-levels ranging from beginner to expert. Currently time constraint prevents us from implementing this feature.
- Multiplayer: A feature to allow the user to play online against other human-players rather than just against the computer. Currently budget and time constraints prevent us from implementing this feature.
- Advertisements: In order to bring in revenue for the game, a feature to be added in the future would be advertisements. These advertisements would be displayed either before the game begins for a set amount of time, or displayed on the sides of the screen. Currently time constraints prevent us from incorporating this.

4.10 Ideas for Solutions

The following points below indicate how we will approach the design/implementations for the code:

- Pygame will be used to implement the checkers game. Although there are various other libraries that could be used, Pygame is perfect for games that are not movement-intensive and for games that do not require high resolution to play.
- Pygame GUI is a useful module that will help us create the GUI. It is open-source and the only dependency is Pygame.

5 Appendix

N/A

5.1 Symbolic Parameters

N/A

References

[1] Wikipedia. Checkers.