SE 3XA3: Software Requirements Specification
T-Rex Acceleration

Team 15, Dev$^{\text{enthusiasts}}$
Zihao Du (duz12)
Andrew Balmakund (balmakua)
Namit Chopra (choprn9)

February 12, 2021

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Version**	**Notes**
02/12/2021	1.0	Initial Draft

This document describes the requirements for the T-Rex Acceleration Game. The template for the Software Requirements Specification (SRS) is a subset of the Volere template (Robertson and Robertson, 2012).

1 Project Drivers

1.1 The Purpose of the Project

The purpose of this project is to modify the simple offline game called T-Rex Runner. The game can be played on the Google Chrome browser when the user is disconnected from the internet. The team will reimplement the game with refreshed graphics and modularized code. The goal is to add new features to a simple game that can be enjoyed by anyone, with or without the internet.

1.2 The Stakeholders

1.2.1 The Client

The clients of this project are the professor and teaching assistants of SFWRENG 3XA3 course.

1.2.2 The Customers

The customers of this project are individuals who enjoy playing 2D computer games and other developers who are interested in developing game modifications (i.e changing different aspects of the game such as its visual or game behaviour).

1.2.3 Other Stakeholders

The developers of the original game (T-Rex Runner) have an invested interest as this project is a redevelopment based on the existing one. Any open source Pygame community will have an invested interest because other developers using Pygame can use this product as a reference in their project. The developers of the product, the members of Team Dev$^{\text{enthusiasts}}$ are stakeholders as well. They will be responsible for the redevelopment of the existing product and future updates.

1.3 Mandated Constraints

- Description: The project shall use the Pygame library to implement the computer graphics.

  Rationale: Pygame contains many GUI modules for game development and is highly portable. It is more efficient to use these Pygame libraries rather than the development team to start from scratch.

  Fit Criterion: All modules have successfully loaded to run the game.

- Description: The project shall be a zero budget project.

  Rationale: The developers of the project have no budget for this project.

  Fit Criterion: The developer team should use open source only.

- Description: The project shall run properly on different operating systems like Windows and Mac OS.

  Rational: The project shall have decent portability.

  Fit Criterion: The user shall be able to run the game on different operating systems.

1.4 Naming Conventions and Terminology

- **SFRWENG 3XA3**: The Software Engineering Practice and Experience: Software Project Management course.
- **T-Rex Runner**: A game that can be played on Google Chrome when the user is disconnected from the internet.
- **GUI**: Graphical User Interface.
- **SRS**: Software Requirements Specification.
- **Pygame**: A Python library composed of modules designed for writing video games.

- **Game state**: Refers to the current game session the user is playing in.
- **OS**: Operating System
- **PEP 8**: The Style Guide for Python Code

1.5 Relevant Facts and Assumptions

1. It is assumed the user knows how to operate a computer.
2. It is assumed the user has a computer that has the capability to run the game.
3. It is assumed the user has physical access to a mouse and keyboard.
4. It is assumed the user has the Python installed version 3.9 or higher.
5. It is assumed the user has Pygame library installed.
6. It is assumed the user has the supporting game files installed on their computer.
7. It is assumed the user can understand basic English.

2 Functional Requirements

2.1 The Scope of the Work and the Product

2.1.1 The Context of the Work

The T-Rex Runner is a 2D endless runner game with black and white graphics and simple features. It can only be played on the Google Chrome browser when the user disconnected from the internet. The redevelopment of T-Rex Runner, called T-Rex Acceleration, will have improved and colourful graphics, including new characters and environment designs. The focus of T-Rex Acceleration is creating a more immersive and addictive gaming experience through new game sounds and features. T-Rex Acceleration will utilize a wider range of game sounds for player movements and obstacles than the original game. The most important aspect of the project to modularize the

game. The project will utilize all free of cost libraries (Pygame), sounds, and GUI assets. Finally, the project will be written in a different programming language, Python instead of JavaScript.

2.1.2 Work Partitioning

Event	Input/Output	Summary
User Controls Character movement	Input: KEYBOARD_JUMP or KEYBOARD_DUCK	System responds and update current game state based on user's input
Collision Detection	Input: Character Position, Obstacle Position Output: Modifying current game state	When the user's character model collides with an obstacle, this will trigger the game state to end.
Change settings	Input: New volumes/ themes selection Output: Modified volume level/themes	The current volume or theme will be changed to the new volume or theme that is selected.
Update Scores	Input: New highest score Output: Updated highest score	When the user's current score is higher than the highest score, the highest score will be updated.
Pause/Resume	Input: User inputs Output: Modified game state	When the user uses RESUME/PAUSE, this will change the state of the game.

Table 2: Work Partitioning Table

2.1.3 Individual Product Use Cases

The primary use case of this product is to play the game from start and trying to keep the character running as long as you can. The following is the use case in detail:

Use case: Play the game
Primary Actor: User

Supporting Actors: None
Precondition: The user has Python and Pygame libraries installed on the computer.
Trigger: The user opens the program
Main Success Scenario

1. User is on the main title screen and starts the game
2. User uses keyboard inputs to dodge obstacles
3. The system counts the current score according to how long the dinosaur runs
4. The user hits on an obstacle and the game ends
5. The system updates the highest score if the current score is higher than the previous ones
6. The user clicks on RESTART to start a new game

Secondary Scenarios

1. User pauses the game: The user presses PAUSE and the game is paused.
2. User resumes the game: The user presses RESUME to continue the stopped game.
3. User leaves the game: The user terminates the game before the player collides with an obstacle. The game will terminate gracefully and the current score will not be recorded.

Success Postcondition: The game is over and the highest score is updated.

2.2 Functional Requirements

1. The character must jump when the user presses KEYBOARD_JUMP as long as the character is on a platform.
   Rationale: The user must able to jump with the character to dodge obstacles. The character must not be able to jump unless on an area meant for running (platform).

2. The character must jump when the user presses KEYBOARD_DUCK as long as the character is on the ground.
   Rationale: The user must able to duck with the character to dodge obstacles. The character must not be able to duck unless on an area meant for running (platform).

3. The system must detect an occurrence of a collision between the character and an obstacle.
   Rationale: When the character model collides with an obstacle (i.e an object meant to impede the movement of the character) the game state must end.

4. The system must spawn different obstacles in random order.
   Rationale: Randomizing the order of the different obstacles spawned, prevents predictable character movement by the user.

5. The system must freeze the current game state when the pause option has been pressed.
   Rationale: It is essential the user has an option to pause the game. The spawning of obstacles, character movement, score tracking, and spawning of power-ups must be paused, preserving the game's state the moment it is paused.

6. The system shall provide a menu for the user to resume the current game or quit the game when the game is paused.
   Rationale: The user must also have an option to resume the game after it has been paused. The user must also have an option to quit the game depending on their preference/situation.

7. The system must exit the current game state and takes the user to the main menu when the quit option is selected.
   Rationale: The user shall be able to leave any time when playing the game.

8. The system must resume back to the current game state with an additional time delay after the resume option has been pressed.
   Rationale: When the user wants to resume back to the current game state, the time delay will allow the user to be aware of the current state of the game. It prevents the user from losing the game instantly by being unaware of the current game situation.

9. The system shall store the score of the current gameplay and display it on the GUI.
   Rationale: The user shall be able to know how they are performing currently in the game.

10. The system shall update the highest score if the current score is higher than the existing highest score.
    Rationale: The system must keep the highest updated allowing the user to reflect on their latest performance.

11. The system must generate different power-ups randomly to be acquired by the user.
    Rationale: Different power-ups will enhance gameplay experience for the user and bring new elements compared to the original game.

12. The user shall be able to acquire a power-up when the character and power-up icon come in contact on the GUI.
    Rationale: The method for the character to obtain the power-up.

13. The character's stats change based on the power-up acquired.
    Rationale: The user should notice a change or get a 'feel' that their character stats have been modified after acquiring the power-up.

14. The system shall play the corresponding sound effects when the player jumps, duck, and collide with obstacles.
    Rationale: The system shall provide an immersive gaming experience.

15. The user must have the option to restart the game or go to the main menu when the game ends (the user hits an obstacle).
    Rationale: The user shall be able to choose if they want to start a new game after the previous game ends. It reduces the amount of time and effort for the user to play another game.

16. The user can select to play the game, change the game settings, and select a different character from the main menu.
    Rationale: The user shall be able to change settings like volume, theme color, and text size in the main menu.

17. The system shall display instructions when the user starts a game
    Rationale: The users need to be guided on how to play when playing for the first time.

3 Non-functional Requirements

3.1 Look and Feel Requirements

3.1.1 Appearance Requirements

LF 1 The viewpoint shall look similar to the original game T-Rex Runner.
Fit Criteria: The game follows the character from a side viewpoint.

LF 2 The menu and game interface shall be properly colored and follow a consistent theme.
Fit Criteria: The menu will contain the use of colors and text in line with the style and art form of the games included.

LF 3 The menu shall be minimalistic.
Fit Criteria: The menu should only contain the essential elements.

3.1.2 Style Requirements

LF 1 The game shall use a bright colour scheme.
Fit Criteria: The color shall contain at least 50% brightness and saturation.

3.2 Usability and Humanity Requirements

3.2.1 Ease of use Requirements

UH 1 The game shall have few and simple controls.
Fit Criteria: Survey a group of individuals and 95% of them should quickly understand the controls of the game within 30 seconds.

UH 2 MINIMUM_AGE and up shall be able to navigate the game with ease.
Fit Criteria: Survey a group of individuals and 95% of them should be satisfied with the ease of use of the game by navigating through all the different menus of the game.

3.2.2 Personalization and Internationalization Requirements

UH 1 The user shall be able to change the volume of the game.
Fit Criteria: The user can choose any volume from 0% to 100%.

UH 2 The user shall be able to change the theme of the game.
Fit Criteria: The user can choose a theme from some provided ones which are constant during the gameplay.

3.2.3 Learning Requirements

UH 1 The user shall get familiar with the game quickly.
Fit Criteria: Survey a group of individuals and 80% of them should understand the basic rules and objectives of the game after their first playthrough.

3.2.4 Understandability and Politeness Requirements

UH 1 The game shall use clear and simple language that can be understood by any English reader above the MINIMUM_AGE.
Fit Criteria: If we put the game instructions into the GRAMMAR_CHECKER, it shall give a readability score that indicates it can be read by the MINIMUM_AGE.

UH 2 The application must use universal symbols for common buttons and functions.
Fit Criteria: Survey a group of individuals and 90% of them should have a positive experience with the user interface.

3.2.5 Accessibility Requirements

N/A

3.3 Performance Requirements

3.3.1 Speed and Latency Requirements

PR 1 The game shall not take more than MAX_DELAY to load all necessary files and libraries.
Fit Criteria: The user must be able to start playing the game before the MAX_DELAY.

PR 2 The game shall respond to the user's input within RESPONSE_TIME.
Fit Criteria: The game must be able to react to the user's input within the RESPONSE_TIME.

3.3.2 Safety-Critical Requirements

N/A

3.3.3 Precision or Accuracy Requirements

PR 1 The score of the user shall be a whole number.
Fit Criteria: The score of the player is displayed on the screen as an integer.

3.3.4 Reliability and Availability Requirements

PR 1 The game shall always be available to download and install most of the time.
Fit Criteria: The user can download and install the game 95% of the time (based on GitLab's availability).

3.3.5 Robustness or Fault-Tolerance Requirements

PR 1 The product shall not fail if unexpected inputs are provided by the user.
Fit Criteria: The game must not crash if non-game control keyboard inputs are provided.

3.3.6 Capacity Requirements

N/A

3.3.7 Scalability or Extensibility Requirements

PR 1 The game shall allow for easy modifications to game features.
Fit Criteria: The developer shall be able to change any visual element in the game.

3.3.8 Longevity Requirements

N/A

3.4 Operational and Environmental Requirements

3.4.1 Expected Physical Environment

OE 1 The game must run on laptops and desktops running the latest version of their respective OS.
Fit Criteria: The user shall be able to run the game on the latest version of Windows, Linux, and Mac OS.

3.4.2 Requirements for Interfacing with Adjacent Systems

N/A

3.4.3 Release Requirements

OE 1 The game's final release shall be before DUE_DATE.
Fit Criteria: The game shall be completed and ready for download on GitLab by this time.

3.5 Maintainability and Support Requirements

3.5.1 Maintenance Requirements

MS 1 The project shall be maintained by developers until DUE_DATE.
Fit Criteria: Any bug discovered before the DUE_DATE shall be fixed as soon as possible by developers.

3.5.2 Supportability Requirements

MS 1 The project shall be supported on different OS's as mentioned earlier.
Fit Criteria: The user shall be able to run the game on the latest version of Windows, Linux, and Mac OS.

3.5.3 Adaptability Requirements

N/A

3.6 Security Requirements

3.6.1 Access Requirements

SR 1 The source code and all game assets must be available for download to everyone.
Fit Criteria: All source code and game assets are available on GitLab.

SR 2 Only the developers and maintainers shall modify the source code and game assets.
Fit Criteria: Only the development team and teaching staff have the role of owners on the GitLab repository.

3.6.2 Privacy Requirements

SR 1 The game must not save any of user's personal information.
Fit Criteria: The product must not access any files and information outside its folder.

3.6.3 Immunity Requirements

SR 1 The developers must not provide any links that may lead to malware.
Fit Criteria: There shall be no URL to an external web page in the project source code.

3.7 Cultural Requirements

3.7.1 Cultural Diversity and Inclusion Requirements

CR 1 The game must not contain any offensive content for people in different cultures.
Fit Criteria: Survey a group of individuals and 100% should be comfortable with the content of the game.

CR 2 The game shall use Canadian English spelling.
Fit Criteria: The in-game text shall be run through the GRAMMAR_CHECKER without any spelling errors.

3.8 Legal Requirements

3.8.1 Compliance Requirements

CR 1 The project must not violate any copyright laws.
Fit Criteria: The project shall abide the BSD-style license with respect to the original source code of the project.

3.8.2 Standards Requirements

LR 1 The code of T-Rex Acceleration must be written using the appropriate coding convention.
Fit Criteria: The code shall follow PEP 8 code style guideline.

3.9 Health and Safety Requirements

N/A

4 Project Issues

4.1 Open Issues

N/A

4.2 Off-the-Shelf Solutions

Any user can play the original game, T-Rex Runner, on Google Chrome when disconnected from the internet. The source code of the original game can provide a reference to implement the basic functionality of the game. As the project will be written using a different language, the original code will be used to implement the fundamental gameplay.

Due to the popularity of the game, there are many variations of T-Rex Runner available online. They are played either on the browser or a smart device. Variations and redevelopments are available on Google Play Store and App Store. There are also open source versions available.

4.3 New Problems

4.3.1 Effects on the Current Environment

The user must have Python 3.9 or higher and Pygame installed to run the game. The game will not require a lot of processing power, thus having little effect on the user's operating system. The user shall not experience any form of lag when running the game.

4.3.2 Potential User Problems

If the user modifies the source code of the project on their local machine, the game may become unstable to play.

4.3.3 Follow-Up Problems

There will be a risk if the libraries, such as Pygame, get updated during the development. This may result in in-game features being unable to run or load.

4.4 Tasks

The following is the link to our project schedule:
Gantt chart and Resource chart

4.5 Migration to the New Product

N/A

4.6 Risks

4.6.1 Testing Risk

Most of the methods in the project will only have a visual output and are highly coupled with other components, thus only limited automated testing can be done. Moreover, obstacles in the game are generated randomly and automated tests are usually insufficient to build confidence for developers. The majority of the testing will be manual system testing. As the game is endless, it is difficult to do exhaustive testing. This increases the probability

of bugs and issues getting undiscovered.
Probability: 20%

4.6.2 Strict Schedule

Due to the strict and tight time constraints, the team must stay disciplined and follow the project schedule. Failure to do so will result in rushed deliverables that are not complete. Rushing the final deliverable will result in incomplete functionality and poor performance.
Probability: 10%

4.7 Costs

This game is based on an open resource game and the development will use free programs and images, so there is no monetary cost for the project.

4.8 User Documentation and Training

4.8.1 User Documentation Requirements

User Documentation for the game will include a ReadME file for the instructions on installation, running the game, and game controls. The game will also include information about the controls and power-ups in the game settings.

4.8.2 Training Requirements

The game controls must be intuitive and simple, thus requires no training.

4.9 Waiting Room

The following features may be added to future versions of the game:

- Multiplayer: To allow multiple users to see and interact with each other in the same game state.
- Local Leaderboard: A leaderboard of the best scores achieved on the local machine.
- Global Leaderboard: A leaderboard of the best scores achieved globally.

4.10 Ideas for Solutions

N/A

References

James Robertson and Suzanne Robertson. *Volere Requirements Specification Template*. Atlantic Systems Guild Limited, 16 edition, 2012.

5 Appendix

This section has been added to the Volere template. This is where you can place additional information.

5.1 Symbolic Parameters

The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.

Table 3: **Symbolic Parameter Table**

Symbolic Parameter	**Description**	**Value**
KEYBOARD_JUMP	Keyboard key that moves the onscreen character vertically upwards	Up Arrow
KEYBOARD_DUCK	Keyboard key that moves the onscreen character duck.	Down Arrow
RESTART	Keyboard key that restarts a new game.	Enter
PAUSE	Keyboard key that pauses the game mid play	Space
RESUME	Keyboard key that resumes the paused game	Space
MINIMUM_AGE	Children younger than this age may have difficulty playing the game	8
GRAMMAR_CHECKER	The engine that checks for spelling errors	Grammarly
MAX_DELAY	The maximum delay time the project should have	5 seconds
RESPONSE_TIME	Typical input delay for the project	5 milliseconds
DUE_DATE	Deadline of the project	04/05/2021