Deliverable #1 Template

SE 3A04: Software Design II – Large System Design

1 Introduction

This is the Software Requirement Specification for Hacker Sim.

1.1 Purpose

The purpose of the document is to lay out a description and intention of the software that is being developed for both the client and the developers. It will convey the software and hardware requirements of the game, as well as the design constraints of the software. Furthermore, this document is used to define our product, its functionality and performance which will be used to verify the product in the testing process.

1.2 Scope

The purpose of the proposed software, Hacker Sim, will be an enjoyable, interactive game that will allow the user to grow their specimen and beat their highscores. The focus of the application is to grow a Software Engineer in their room. Both the Software Engineer and their room can be upgraded using the in-game currency. The in-game currency will be acquired through the completion of tasks. Online capabilities will give users the ability to share their rooms and chat as well as give gifts to other users to help complete tasks.

1.3 Definitions, Acronyms, and Abbreviations

SE - Software Engineer

1.4 References

Dormehl, L. (2019, May 29). The Tamagotchi Effect: How Digital Pets Shaped The Way We Use Technology. Retrieved from `https://www.digitaltrends.com/cool-tech/how-tamagotchi-shaped-tech/`

Technologies, U. (n.d.). Unity. Retrieved from `https://unity.com/`

1.5 Overview

The document is organized in the following manner: Overall Description, Use Case Diagram, Functional Requirements, Non-Functional Requirements. Overall Description focuses on providing details about how the ideas link together to enable specific functionality and showing functionality through comparison as well as outline who the users are intended to be and the constraints. Use Case Diagrams demonstrate how the stakeholder carries out business events. Functional Requirements explains the functionality of the system. Finally, the Non-Functional Requirements focuses on the required qualities of the software.

2 Overall Description

2.1 Product Perspective

The product is a system modeling application for desktop computers based on the Unity platform using C#. The product design itself is not a continuation of any predecessor products, but in the past there were many implementations with a similar concept; raising a digital being. The trend of virtual pet games started from mid 1990s to early 2000s, popularized by the Tamagotchis. Our product will differentiate itself by giving users an experience of raising a software engineer from the very beginning.

The system's main features:

1) A server with functionalities:
   - To provide users with online access via login/logout
   - To connect a database with the implementation of a game
   - To allow different users to share and interact with each other
   - To allow users to give feedback and report bugs
2) A database or a file management system on the user's computer for storing:
   - User account information set
   - User game progress
   - Product FAQs for convenient look-up

2.2 Product Functions

2.2.1 Function Summary

The system's functionality is centralized around the user raising their digital specimen as a Software Engineer (referred to as SE below). The user will be involved with the lifecycle of specimen and attempt to promote it with a set of means and reach end-game goal. A currency system is designed for the in-game shop and the user can earn currency from initiating interactions/events with their SE. For example, working and completing coding projects. The in-game shop contains three main categories of items: interactable items (pets and exercise equipment), furniture and room upgrades and food. The specimen has a list of metrics, i.e. tiredness, happiness, age, health, etc. The user must follow certain rules to keep metrics within a rational range, otherwise it will hinder specimen growth or lead to game failure. The user will be able to view their specimen's information to check their progress and current health.

2.2.2 List of Basic Functions

1. User creates a SE.
2. Users initiate events and interact with their SE.
3. Users purchase items from the in-game shop.
4. Users utilize items.
5. Users customize specimen's room.
6. Users view specimen's information.
7. Users reset their game stage and start a new one.

2.2.3 Function Domain

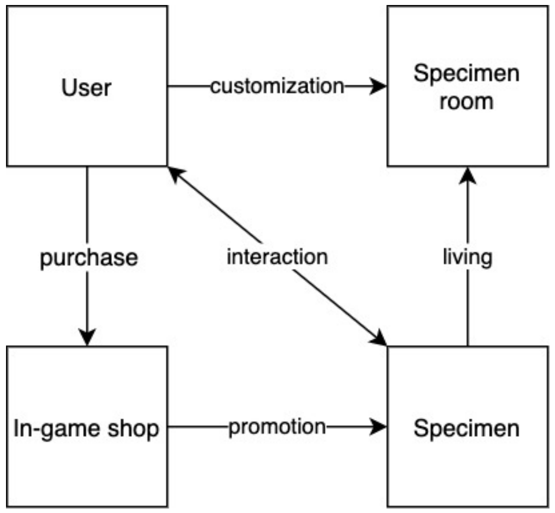

Figure 1: Function Domain

2.3 User Characteristics

a) Describe those general characteristics of the intended users of the product including educational level, experience, and technical expertise

b) Do not state specific requirements, but rather provide the reasons why certain specific requirements are later specified

2.4 Constraints

a) Provide a general description of any other items that will limit the developer's options

2.5 Assumptions and Dependencies

a) List each of the factors that affect the requirements stated in the SRS

b) These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS

- **Example**: An assumption may be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.

2.6 Apportioning of Requirements

a) Identify requirements that may be delayed until future versions of the system

3 Use Case Diagram

This section should provide a use case diagram for your application.

a) Each use case appearing in the diagram should be accompanied by a text description.

4 Functional Requirements

This section of the SRS should contain all of the software requirements to a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements. Throughout this section, every stated requirement should be externally perceivable by users, operators, or other external systems. These requirements should include at a minimum a description of every input (stimulus) into the system, every output (response) from the system, and all functions performed by the system in response to an input or in support of an output.

You normally have two options for organizing your functional requirements:

1. Organize first by *business events*, then by *viewpoints*
2. Organize first by *viewpoints*, then by *business events*

Choose the one which makes the most sense.

For example, if you wish to organization by business events:

BE1. Business Event

- VP1.1 Viewpoint
  - i. Requirement
  - ii. Requirement
  - iii. . . .
- VP1.2 Viewpoint
  - i. Requirement
  - ii. Requirement
  - iii. . . .
- VP1.3 . . .

BE2. Business Event

- VP2.1 Viewpoint
  - i. Requirement
  - ii. Requirement
  - iii. . . .
- VP2.2 Viewpoint
  - i. Requirement
  - ii. Requirement
  - iii. . . .
- VP2.3 . . .

<u>OR</u>, if you wish to organization by viewpoints:

VP1. Viewpoint

- BE1.1 Business Event
  - i. Requirement
  - ii. Requirement
  - iii. . . .
- BE1.2 Business Event
  - i. Requirement
  - ii. Requirement
  - iii. . . .

BE1.3 . . .

VP2. Viewpoint

BE2.1 Business Event

i. Requirement
ii. Requirement
iii. . . .

BE2.2 Business Event

i. Requirement
ii. Requirement
iii. . . .

BE2.3 . . .

5 Non-Functional Requirements

5.1 Look and Feel Requirements

5.1.1 Appearance Requirements

LF1.

5.1.2 Style Requirements

LF1.

5.2 Usability and Humanity Requirements

5.2.1 Ease of Use Requirements

UH1.

5.2.2 Personalization and Internationalization Requirements

UH1.

5.2.3 Learning Requirements

UH1.

5.2.4 Understandability and Politeness Requirements

UH1.

5.2.5 Accessibility Requirements

UH1.

5.3 Performance Requirements

5.3.1 Speed and Latency Requirements

PR1.

5.3.2 Safety-Critical Requirements

PR1.

5.3.3 Precision or Accuracy Requirements

PR1.

5.3.4 Reliability and Availability Requirements

PR1.

5.3.5 Robustness or Fault-Tolerance Requirements

PR1.

5.3.6 Capacity Requirements

PR1.

5.3.7 Scalability or Extensibility Requirements

PR1.

5.3.8 Longevity Requirements

PR1.

5.4 Operational and Environmental Requirements

5.4.1 Expected Physical Environment

OE1.

5.4.2 Requirements for Interfacing with Adjacent Systems

OE1.

5.4.3 Productization Requirements

OE1.

5.4.4 Release Requirements

OE1.

5.5 Maintainability and Support Requirements

5.5.1 Maintenance Requirements

MS1.

5.5.2 Supportability Requirements

MS1.

5.5.3 Adaptability Requirements

MS1.

5.6 Security Requirements

5.6.1 Access Requirements

SR1.

5.6.2 Integrity Requirements

SR1.

5.6.3 Privacy Requirements

SR1.

5.6.4 Audit Requirements

SR1.

5.6.5 Immunity Requirements

SR1.

5.7 Cultural and Political Requirements

5.7.1 Cultural Requirements

CP1.

5.7.2 Political Requirements

CP1.

5.8 Legal Requirements

5.8.1 Compliance Requirements

LR1.

5.8.2 Standards Requirements

LR1.

A Division of Labour

Include a Division of Labour sheet which indicates the contributions of each team member. This sheet must be signed by all team members.

IMPORTANT NOTES

- Be sure to include all sections of the template in your document regardless whether you have something to write for each or not
  - If you do not have anything to write in a section, indicate this by the *N/A*, *void*, *none*, etc.
- Uniquely number each of your requirements for easy identification and cross-referencing
- Highlight terms that are defined in Section 1.3 (**Definitions, Acronyms, and Abbreviations**) with **bold**, *italic* or underline
- For Deliverable 1, please highlight, in some fashion, all (you may have more than one) creative and innovative features. Your creative and innovative features will generally be described in Section 2.2 (**Product Functions**), but it will depend on the type of creative or innovative features you are including.