SE 3A04: Software Design III: Large System Design

Group #5, Spaceship System Sabatoge
Pareek Ravi 001407109
Pavle Arezina 001410366
David Hobson 001412317
Victoria Graff 001401451
Julian Cassano 001406891

February 12, 2017

Contents

List of Tables

List of Figures

Table 1: **Revision History**

1 Overall Description

1.1 Product Perspective

Spaceship System Sabotage is an independent and self-containted system. The typical user experience with this application is to keep the system operating for as long as possible so that after a specified period of time the space craft will "reach its intended destination" and the user will reach an end state through a win condition. This perspective on the interaction of sub-systems will make the application feel more like a game than it does a simulation. When a sub-system is inputted with a stimulus to cause it to fail, the application will tell the user that one of the crew members of the ship has sabotaged a sub-system. This acts as a challenge for the user to keep the crew members alive but attempt to determine who the defector of the crew is, providing "lore" and context for the simulated sub-system failure so that the user has motivation to keep the system functional.

1.2 Product Functions

- The user will be able to start a simulation of a system
- The different sub-systems will be displayed for the user
- The user will be able to switch views to choose the sub-system they are observing
- The system will allows the user to combine views
- Messages/alerts will be displayed for the user
- The sub-systems will be able to receive stimulus from the application to prompt a status change
- The sub-systems will be able to affect other sub-systems depending on their status
- The user will be able to input a stimulus that will have a specified effect on a sub-system, namely a status change back to the original state
- The user will be able to pause/resume the system
- The simulation will be able to reach an end state at any time and terminate the simulation

- The user will be able to end the simulation at any time if they so choose

1.3 User Characteristics

The intended user of this application is anyone who enjoys games or challenges and has a basic understanding of systems and how they can interact. No previous work or school experience with designing systems is required as this application is not a test on the user's understanding of software architecture. This application aims to provide a casual experience of simulating the negative effects of sub-systems failing in a larger system and allows the user to have a role in this scenario regardless of failure or not. Each end state is an acceptable level of completion in terms of the simulation as they both result in a demonstration of the interaction between sub-systems. This means that the application is accessible to anyone from casual players in elementary school to Dr. Khedri while he's killing time waiting for his TAs to mark these deliverables.

1.4 Constraints

The limitations of this project come in the form of development time. The allocated amount of hours that we intend to dedicate to this project are in negative correlation with the amount of other school work we have to do during the timeframes of each deliverable. We would love to make this a full fleshed out game that is fun and challenging but at the moment we do not have the time to comfortably meet the requirements of the project and introduce enough additional features to make this a fun to play, sellable game. The rubric specified by the TAs will first and foremost be followed, and only then can additional features be added.
Another limitation of this project would be the programming language chosen for the application. Depending on what language the development takes place in, certain features would be more difficult to implement in the amount of time given. For example, our team is more experienced in Java and may be more proficient in meeting the functional requirements with this language but it may be harder to meet the non-functional requirements, especially in terms of visual interfacing compared to working in a language like C++.

1.5 Assumptions and Dependencies

The assumptions in the requirements section of this document are focused around the intended interface of the application. The proposed layout of the visual interface is

that the sub-systems will be laid out in a representation of their location in physical space, meaning that there will appear to be a space craft in the user's view and all of the stimulus will be inputted through this apparent view. This also means that sub-systems will be layered on top of each other and will be difficult to view all at once, hence the requirement that the uer will be able to change views and customize the sub-systems being displayed at one time.

1.6 Apportioning of Requirements

Some non-functional requirements that could take too long to implement and are not explicitly required for a functional demonstration may be delayed until later versions of the application. Some areas of development that this is a concern for are the low loading times and "clean" interface.

2 Non-Functional Requirements

2.1 Look and Feel Requirements

2.1.1 Appearance Requirements

LF1. The application shall have a clean, minimalistic user interface. The systems on the ship will be colour coded and easy to differentiate on the map.

2.1.2 Style Requirements

LF1. The application shall have smooth transitions while responding to user input.

2.2 Usability and Humanity Requirements

2.2.1 Ease of Use Requirements

UH1. The application shall be simple to use for all users above the age of twelve. A new user shall be able to navigate through menus and play the game with ease.

2.2.2 Personalization and Internationalization Requirements

UH1. There will be no personalization or Internationalization requirements.

2.2.3 Learning Requirements

UH1. The learning time within the application shall take no longer than five minutes.

2.2.4 Understandability and Politeness Requirements

UH1. The application will be easy to understand and will refrain from inappropriate language or slang words.

2.2.5 Accessibility Requirements

UH1. The application will be playable for the hearing impaired.

2.3 Performance Requirements

2.3.1 Speed and Latency Requirements

PR1. The application shall update systems from user input in under 5 seconds, and load time between screens shall be under 5 second.

2.3.2 Safety-Critical Requirements

PR1. There will be no Safety-critical requirements.

2.3.3 Precision or Accuracy Requirements

PR1. The touch user input on the application will have 90 percent accuracy.

2.3.4 Reliability and Availability Requirements

PR1. The application shall be available at all times.

2.3.5 Robustness or Fault-Tolerance Requirements

PR1. The application will be designed to handle all selection errors by the user.

2.3.6 Capacity Requirements

PR1. The application will be able to handle input from one user.

2.3.7 Scalability or Extensibility Requirements

PR1. The application will be created to be able to integrate more space ship systems in the future.

2.3.8 Longevity Requirements

PR1. The application will operate as long as the Android OS version it is created on, is supported.

2.4 Operational and Environmental Requirements

2.4.1 Expected Physical Environment

OE1. The application will run on Android smartphones.

OE2. The application shall be used at any location.

2.4.2 Requirements for Interfacing with Adjacent Systems

OE1. There will be no adjacent systems used through this application.

2.4.3 Productization Requirements

OE1. There will be no productization requirements for this application.

2.4.4 Release Requirements

OE1. The application will be downloadable through the Google Play Application store.

2.5 Maintainability and Support Requirements

2.5.1 Maintenance Requirements

MS1. The application shall be built through modular design for easy maintenance and for updating features in the future.

2.5.2 Supportability Requirements

MS1. The application will only run on android smartphones.

2.5.3 Adaptability Requirements

MS1. The application will be designed to be able to re-format to fit all android phone screen sizes.

2.6 Security Requirements

2.6.1 Access Requirements

SR1. The application will be availability to users that own an android smartphone.

2.6.2 Integrity Requirements

SR1. No personal information will be required.

2.6.3 Privacy Requirements

SR1. No personal information will be required.

2.6.4 Audit Requirements

SR1. There will be no Audit requirements for this application

2.6.5 Immunity Requirements

SR1. There will be no immunity requirements for this application.

2.7 Cultural and Political Requirements

2.7.1 Cultural Requirements

CP1. The application will display language in the english language and will avoid mention or assumption of any specific culture.

2.7.2 Political Requirements

CP1. The application will not mention any political topics or issues.

2.8 Legal Requirements

2.8.1 Compliance Requirements

LR1. The application shall adhere to all compliance laws in Ontario, Canada.

2.8.2 Standards Requirements

LR1. The application shall adhere to all standards laws in Ontario, Canada.