Software Requirements Specification

for

Astropolis Mini-Game

Version 1.0

Prepared by Team Astropolis

Team Members: Robert Larivee, Phillip Levinson, William Wood

July 24, 2009

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Robert Larivee	7/24/09	Initial Draft	1.0

1. Introduction

1.1 Purpose

This SRS is for a video game which will be used to test people with Autism spectrum conditions. This game will be embedded inside of a larger game, called Astropolis, which contains a number of other mini-games.

1.2 Document Conventions

N/A

1.3 Intended Audience and Reading Suggestions

This document should be read by the developers, customers, and Faculty advisor. The developers should read every section to ensure that there is an understanding for the project. The main sections for the customers and faculty advisor to review are section 1.4 Project Scope, 2.7 Assumptions, and section 3. Features.

1.4 Project Scope

The purpose of this software can be broken down into two major goals. The first is to allow experimental data to be collected from the players of the game. This data is valuable in the research of Autism Spectrum conditions. The second goal is to have a fun game that actually helps people with Autism Spectrum conditions to recognize facial expressions. This goal is important because many people with Autism spectrum conditions have difficulty in recognizing emotions based on facial expressions.

1.5 References

None

2. Overall Description

2.1 Product Perspective

The Astropolis Mini-game will be a new addition to Astropolis, which is a colony simulator that contains a number of mini-games. The new mini-game will rely on some previously implemented components and resources of the Astropolis game.

2.2 Product Features

3. The mini-game will display faces with varying expressions. These faces will be images of people which can be easily replaced by the customer.

1. Players will be awarded points for matching a face portraying a certain emotion with other faces showing the same emotion.
2. Players will be awarded credits for the Astropolis colony simulator after completing the mini-game.
3. All of the player's moves and game events will be logged into a file with a timestamp.
4. Multiple levels will exist, with varying degrees of difficulty.

3.1 User Classes and Characteristics

The set of users who will most frequently use this software will have an Autism spectrum condition. These conditions vary greatly, but the main user audience will consist of high-functioning Autistic spectrum users. The target demographic will be children and young adults, but the experimental data collected would be valuable from any age group.

3.2 Operating Environment

The Astropolis mini-game will require the XNA 2.0 redistributable in order to run on a client's machine. This software can only be run in Microsoft Windows XP or greater. When a test is being executed in the lab there will be additional hardware, including an EEG reader. This additional hardware is not integrated with the game, and will run independently.

3.3 Design and Implementation Constraints

The main design constraint is that the game must be playable by people with Autism spectrum conditions, therefore the proper considerations must be made for the users. There are certain safety requirements that we must follow. See section 5.2.

The mini-game will be written in C# using the XNA libraries and functions. Therefore, the mini-game will comply to the programming practices of C#. The delivered software will be open-source and maintained by Belmonte Labs.

3.4 User Documentation

There will be a basic tutorial document along with an in game tutorial to aid users.

3.5 Assumptions and Dependencies

1. The Astropolis event logger component will be fully functional and available to the mini-game.
2. XNA 2.0 remains stable and compatible with Windows XP and greater.
3. The Astropolis game will be functional.

4. System Features

4.1 Face Submission

3.1.1 Description and Priority

The customer and end-users should be able to easily up-load an image into the game to be used. When uploading an image, the user should be able to specify which category of emotion that the picture belongs to. This task is of Low priority.

3.1.2 Stimulus/Response Sequences

When starting the mini-game, the user will be given the option on the main menu to add faces to the game. This will load an interface that will as for the path to the desired picture and the emotion of the picture. The system will then save the image into a directory based on the emotion of the face.

3.1.3 Functional Requirements

REQ-1: A picture must be added to the game.
REQ-2: A picture must be associated with the correct emotion.

4.2 Facial Expression Matching

3.2.1 Description and Priority
Users will be rewarded points for identifying which faces are missing from the game board. Refer to section 3.2.2 for more details. This feature is High priority.

3.2.2 Stimulus/Response Sequences

The users should be presented a game board of faces in a grid. See Appendix for screenshot. Certain faces will be removed from the grid and lined up at the bottom of the screen. The user must select which missing face will create a row of 3 identical emotions in the game board.

3.2.3 Functional Requirements

REQ-1: A game board will exist in a 3x4, 4x4, or 4x5 grid.
REQ-2: The game board will be filled with faces depicting emotions.
REQ-3: Random faces will be removed from the board and replaced with a blank space.
REQ-4: Faces will appear in a row at the bottom of the screen outside of the grid one-by-one at an interval of 1 second +- a 200 millisecond jitter.
REQ-5: Players can place a face back on the game board by clicking on the face and then clicking on the blank space on the board.
REQ-6: Players can remove faces they have placed by clicking on the face and pressing a remove button.
REQ-7: Players will press a button labeled “Submit” to begin validation of the game board.
REQ-8: Players can press “Submit” with one or more blank space filled with their answer.

REQ-9: Pressing "Submit" will validate the board by checking to see if the filled in spaces connect 3 like emotions.
REQ-10: After pressing "Submit", if the user-filled blank space does not connect 3 like emotions horizontally or vertically, then the answer was incorrect and the face is removed from the board and placed back at the bottom of the screen outside the grid.
REQ-11: If the answer was incorrect then the player looses points.
REQ-12: After pressing "Submit", if the user-filled blank space connects 3 like emotions horizontally or vertically then the player will be rewarded points.
REQ-13: After pressing "Submit", bonus points will be given if the player gets multiple correct answers.
REQ-14: There will always be faces available that will successfully connect 3 like emotions.

4.3 Credits Given For Astropolis Colony Sim

3.3.1 Description and Priority

The completion of the mini-game will result in a reward for the player which can be used in the Astropolis colony simulator. This reward, along with the players score will be shown after the mini-game is completed on an End Game screen. Medium Priority.

3.3.2 Stimulus/Response Sequences

The stimulus would be the completion of the min-game. The user is presented an End Game screen and must press the 'Space-bar' to exit back to Astropolis.

3.3.3 Functional Requirements

REQ-1: A screen is presented at the mini-games completion which shows score and statistics.
REQ-2: A reward will be given which can be used in the colony simulator based on performance in the mini-game.
REQ-3: Users press 'Space-bar' to exit back to Astropolis from the End Game screen.

4.4 Events Logged

3.4.1 Description and Priority

All of the events in the game are logged into a file with the event name and time stamp. High Priority.

3.4.2 Stimulus/Response Sequences

Events from the system and user will be logged constantly. Events include: The appearance of the faces, the selection of faces, the placement of faces, etc.

3.4.3 Functional Requirements

REQ-1: Each log entry must have a timestamp.
REQ-2: All events are logged in a log file.

4.5 Multiple Levels

3.1.1 Description and Priority

The mini-game will consist of multiple levels. At the completion of a level, another board will be generated. Each new board generated will be slightly more challenging by adding new rows or columns to the board, adding more possible faces to fill-in the blank spaces with, or by adding emotions to the board that have subtle differences. This task is of High priority.

3.1.2 Stimulus/Response Sequences

Each level will be randomly generated when the new level is started.

3.1.3 Functional Requirements

REQ-1: At the completion of a level, a new level will be generated.
REQ-2: At the completion of the 5th level, the game will end.
REQ-3: More points will be given for more difficult levels.

5. External Interface Requirements

5.1 User Interfaces

The look and feel must match the current Astropolis look and feel and must capture part of the space theme prevalent throughout the program. Other then these specific constraints the UI we have free reign as long as we also are cautious of using autistic friendly UI designs. Any instruction will have to be visual and be concise complicated procedures will prove difficult for Autistic users.

5.2 Hardware Interfaces

There are 2 interfaces for us to interact with keyboard and mouse. Both of these are handled by the XNA classes and will mostly be simple for us to interact with. The program requires rather modest computer hardware.

5.3 Software Interfaces

The game currently only supports the windows operating systems (XP, Vista, Windows7). There will be two software interfaces one between our game and the overlying Astropolis program and then between our game and the ECS monitoring software. This first one will be mostly the sharing of some underlying code and DLL's. The second is much more loosely coupled as we just write events out to a interface and that will handle the monitoring of the player.

5.4 Communications Interfaces

There aren't any communication interfaces as currently the game is single player and does not send out the data it collects unless run in the lab.

6. Other Nonfunctional Requirements

6.1 Performance Requirements

The game must be able to run at 50-60 frames per second on Windows platform OS. At this frame rate, the game will remain constant and playable. It will not be technically demanding and able to run on lower end computers.

6.2 Safety Requirements

The main safety concern that we will be dealing with is the possibility of inducing seizure in the user. Some autistic children are very susceptible to flashing lights and this can be problematic. To safeguard against this we will try to keep flashing lights and rapid screen movements down to a minimum. We will not flash the screen or change colors rapidly when it is unnecessary to do so.

6.3 Security Requirements

Security will not be a concern for this project since there is no sensitive data being stored.

6.4 Software Quality Attributes

The two main quality attributes that the astropolis mini-game will be focusing on is correctness and usability. The system needs to be able to track time correctly. Since the mini-game will need to record events and timestamp them with extremely high accuracy, it is important that the timing is accurate and consistent. If the of these events were to be off by a small amount then the research data provided could be incorrect.

Usability is also a priority because this is a game. If the user were to be confused or annoyed by the User interface, then the game would be less enjoyable. There are also special considerations for autistic children to include when developing the UI. These are more detailed on the autisim collab wiki.

7. Other Requirements

None

Appendix : Models