Indecent Exposure

Software Requirements Specification

CIS 422

Spring 2016

Portia Seater

Charlie Quinn

Alex Brandenfels

1. Introduction

The purpose of our game is to provide light entertainment that will amuse audiences of different ages and experiences. We believe the theme is easily relatable and will garner interest from potential users without much difficulty.

1.1 Intended Audience and Purpose

This game is intended for the current developers and anyone who wishes to continue development in the future.

The purpose of the document is to concatenate relevant project information into one main source. Although this document is not large enough to contain all of the details of the project design and specifications, it also serves to indicate where that information can be found.

1.2 How to use the document

This document serves as an overview of the technical specifications of the project. Below is an index of the content.

1. Introduction and intention of document

2. Concept of Operations

3. Behavioural Requirement

4. Quality Requirements

5. Fundamental Assumptions

6. Expected Changes

7. References

8. Appendices

2. Concept of Operations

This game will provide our users with their daily dose of the absurd. The premise of the game plays off the idea of being trapped in a nightmare in which you’re completely naked.

Game play starts with the user entering their dream world and realizing they are completely without clothing. The goal is to avoid being seen by peers and colleagues while searching for items of your clothing in a multilevel map of a high school. We plan on building three levels of the game, which correspond to the three floors of the school. Your character will follow a fairly linear path and use objects such as desks to hide behind.

To see our extended Concept of Operations document, please visit the wiki page. There you will be provided with a more in-depth background behind the project.

2.1 System Context

This system will be created using the SDL2 cross-platform development library. It is used to create games, video players, and emulators and is available under the zlib license. This will be included in the installation for users, so users will not need to have SDL2 independently installed.

The user does not need to create any kind of account to play the game. After first installation there will not be any steps besides running it.

The game will be compatible with both the Windows and Mac OS.

2.2 System Capabilities

* A splash screen will act as home screen of the game. It will be intiated upon startup.
* In game experience will include:
  + Window displaying level layout
  + Written text displaying self-updating game time
  + Written text displaying number of times player has been “seen” by NPC’s
  + Ability to pause game. This initiate a pause screen that is visible until the game is restarted.
  + Winning screen for when player beats level.

3. Behavioral Requirements

3.1 System Inputs and Outputs

3.1.1 Inputs

Keystrokes:

Space Bar:

Allows user to begin game-play

Q:

Allows user to quit game and close window

P:

Allows user to pause game play

Up:

Allows user to move character towards the top of the game screen

Down:

Allows user to move character towards the bottom of the game screen

Left:

Allows user to move character towards the left of the game screen

Right:

Allows user to move character towards the right of the game screen

3.1.2 Outputs

Main game screen:

Updates each frame with new status of NPC and main player animation, as well as any collison detection with characters, walls, or items.

Character Movement:

Updated each frame to reflect the user directional input.

Pause Screen:

Appears in place of main game window when game is in a paused state to signify state to user.

Winning Screen:

Appears in place of main game window when user has beaten the level to signify the game is over.

Game time:

Appears in upper corner of main game window and is refreshed each frame to reflect the total game play time for that level.

Times Seen:

Appears in upper corner of main game window and is refreshed each frame to reflect the total number of times the player has been “seen” so far in current game play.

3.2 Detailed Output Behavior

Use Case Models

3.2.1 Use Case: Launch Game Executable

* Actors: User
* Preconditions: User has downloaded game and has a keyboard available.
* Triggers: User has just opened the game, which then runs.
* Flow of Events:
  + Basic Flow:

1. Game opens to first splash screen.
2. User reads text on screen.
3. User presses the ‘Spacebar’ to continue to game.
4. The instructions splash screen is displayed.
5. User reads the instructions.
6. User presses the ‘s’ key to continue.
7. Game play begins.
   * Alternative Flows:

(Steps 1 and 2 executed)

3. User presses the ‘esc’ key to quit.

4. Window closes and the game exits.

* Exceptions: User clicks the ‘close’ title bar button and the game exits.
* Post Conditions: Game play has begun, or the window has been closed.

3.2.2 Use Case: Game Play Begins

* Actors: User
* Preconditions: User has run game and has just moved past the opening splash screens into game play.
* Triggers: User has just hit the ‘s’ key to start the game.
* Flow of Events:
  + Basic Flow:

1. User observes the layout of the level.
2. User sees the NPC’s moving and observes their paths
3. User presses the up arrow key
4. The character sprite moves up towards the top of the window
   * Alternative Flows:

(Steps 1 and 2 executed)

3. User presses the up arrow key

4. The character sprite moves up towards the top of the window

(Steps 1 and 2 executed)

3. User presses the down arrow key

4. The character sprite moves down towards the bottom of the window

(Steps 1 and 2 executed)

3. User presses the right arrow key

4. The character sprite moves towards the right of the window

(Steps 1 and 2 executed)

3. User presses the up arrow key

4. The character sprite moves towards the left of the window

* Exceptions:
  + User chooses not to press any arrow keys. Character sprite does not move.
  + User presses keys besides the arrow keys. Character sprite does not move.
* Post Conditions: The user has moved the main naked character sprite either up, down, left, or right.

3.2.3 Use Case: Main Character is Seen

* Actors: User
* Preconditions: User has entered game play successfully and has made at least minimal movement.
* Triggers: Main character sprite is seen from an NPC detecting it via the line of sight algorithm.
* Flow of Events:
  + Basic Flow:

1. Character intercepts a NPC’s line of site.

2. Character sprite is moved back to start of level map.

* + Alternative Flows:

None

* Exceptions: If the character is on the other side of a wall from the NPC, the line of sight detection will not detect them.
* Post Conditions: Character is once more at the start of the level map. User is frustrated and challenged to try again.

3.2.4 Use Case: Item Pickup

* Actors: User
* Preconditions: User has entered game play successfully and has moved to a section of the level that has an item.
* Triggers: Main character moves towards item.
* Flow of Events:
  + Basic Flow:

1. User presses the appropriate arrow keys to navigate character towards item.

2. Character collides with item.

3. Collision is detected.

4. Item sprite no longer appears on game screen.

* + Alternative Flows:

(Steps 1 through 4 executed)

5. Item was a clothing item.

6. Character sprite is redrawn wearing clothing item.

(Steps 1 through 4 executed)

5. Item was a bonus item.

6. Character gains the given bonus for a limited number of seconds.

7. Gameplay returns to normal.

* Exceptions: Item is final item at end of level. Enter ‘End of Level’ use case.
* Post Conditions: Character sprite has changed, or character is granted temporarily elavated abilities.

3.2.4 Use Case: End of Level

* Actors: User
* Preconditions: User has entered game play successfully and has moved through the level in its entirety.
* Triggers: Main character collides with final clothing item in fornt of exit door.
* Flow of Events:
  + Basic Flow:

1. User presses the appropriate arrow keys to navigate character towards item.

2. Character collides with item.

3. Collision is detected.

4. Splash screen appears.

5. User congratulates themselves.

6. User presses the ‘c’ key to continue to the next level.

7. The next level replaces the splash screen in the window.

* + Alternative Flows:

(Steps 1 through 5 executed)

6. User presses the ‘q’ key to quit the game.

7. The game window is closed.

(Steps 1 through 5 executed)

6. The user has beaten all levels.

7. User presses the ‘q’ key to quit the game.

8. The game window is closed.

* Exceptions: User clicks the ‘close’ title bar button and the game exits.
* Post Conditions: The player has either advanced to the next level, or quit the game.

4. Quality Requirements

Consistent Game Play

Intuitive Design

Challenging and Interesting

6. Fundamental Assumptions

We assume that there will be no changes to the SDL2 development library. We also assume the user has basic familiarity with either the Windows or Mac OS.

Other assumptions are that the user has played a video game before and is familiar enough to infer the basic goals in a maze style sneaking game from limited instruction.

7. Expected Changes

We expect:

* The list of features to fluctuate throughout development. These will depend on the difficulty of implementation, time taken to create any relevant art, and time remaining before the project deadline.
* Art to change, with new art being developed and old art being revised.
* Level design to change as we create new features and experiment to see what is engaging and what is not.
* There may be addtions of new levels later in the project timeline, or by future developers looking to expand the game.
* Documentation will change as more features are added and redacted, or if future developers continue work after the project deadline.

8. Appendices

8.1 Definitions and Acronyms

8.1.1 Definitions

Sprite: A two-dimensional independent object that represents things like an item or person

Isometric: A method of visually representing three-dimensional objects in two-dimensions.

Frame: An image that is temporarily displayed in the game window, and is updated at least 24 times per second, usually more with current technology.

zlib license: Quoted from <http://www.gzip.org/zlib/zlib_license.html>:

This software is provided 'as-is', without any express or implied warranty. Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not

claim that you wrote the original software. If you use this software

in a product, an acknowledgment in the product documentation would be

appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be

misrepresented as being the original software.

3. This notice may not be removed or altered from any source distribution.

8.1.2 Acronyms

SDL2 : Simple DirectMedia Layer, version 2

NPC: Non Player Character

Concept of Operations

Purpose and Audience

This document is intended to be a general overview of the project for those who are interested in the premise and motivations behind creating the game. It is intended to provide more insight to users who want background on the development, or prospective developers who would like to get a better idea of the concept of the project before moving into more technical documentation.

Project Concept Motivation

We set out to create a game that would be simple at it’s heart, but rich enough to entertain a wide audience. We were inspired by the simplicity of premise behind many 8-bit and iternet based games, and knew that we would be able to achieve worthwhile results by following that path. The idea to build off of the classic “naked at school nightmare” fit all of our criteria and we believe it to be a fear that is relatable to almost everyone. Our implementation aims to reflect this mood of playfulness in its 8-bit aesthetic, which also helps to keep the nudity as a narrative device rather than shock factor.

Value

We aim to create a piece of software that will provide a few laughs, as well as some moderately challenging game play. We feel that the relatability of this idea will make the game successful, as unplanned public exposure is a fear that can be understood at all ages. Although this game may sound trivial, there is consumer demand for the genre. Simple, comedic games are fairly popular (Shower With Your Dad Simulator 2015, Surgeon Simulator 2013, Goat Simulator, Who's Your Daddy, etc.).

Product Scope

Our game will be made available as a cross-platform application for PC users. We won’t be implementing it for use with any other consoles. We plan on using the SDL game development library for our game engine. This will mean using SDL2 and SDL2 image libraries in our code. Our program will be written in C++.

Main Dependencies

Before beginning to design our game we will need a C++ program that compile with SDL libraries without errors. This requires us to have SDL installed and set up properly. Then we can begin building the tools for our game, namely sprites and the controls to our gameplay. After that we will design a level to test with, and then we can work on a function-based AI. When we have working NPCs that can keep the game competitive we will design a few levels required to beat the game.

Design Concept

Architectural Design Sketch Our game will be an object-oriented program. Our main class will initialize our biggest classes. This will include our SDL\_setup class that will handle all events, like I/O, and the rendering of the game's window. Our main class will also contain a function named GameLoop() that will run every frame. The gameloop will update an Environment object, that will contain all aspects of what is displayed on the screen. We will use sprite objects to manipulate images on the screen, and some sort of collision object will be a part of our Sprite class.

Project Plan

1. Purpose and Audience

This document is intended mainly for developers of the Indecent Exposure game. Anyone who is interested in the development of the game may also find this to be an interesting document.

The document serves as a way to synthesize the team plans and keep the team on track for the deadline. Team members can review their role and check to make sure we are dealing with risks as we expected to, and if we mispredicted something in our initial planing stages.

2. Team Roles

Alex Brandenfels

Developer

Tester & Integrater

Charlie Quinn

Architect

Developer

Portia Seater

Game Art

Level Design

Features Implementation

Documentation

3. Risks and Mitigation

Project is not completed by deadline…

Creating a program that’s fun for users…

We will design with game difficulty in mind, and take from our own experiences playing games what is and is not fun. We will have test users run through our game and take their feedback into account.

Successfully implementing collision detection…

Successfully implementing line of sight detection…

Not having enough time to complete features…

We will priortize features that are vital to game play over others that more of reach goals. This will enable us to have the solid foundation of a MVP that we can continue to build off of as we have time. If there is time for extra features, we will start by working on ones we know will take less time to create.

Not having enough time to create all of the levels…

We will make as many as we are able to in the time given, and if we don’t meet our goal we will increase the difficulty of an existing level to create at least a bit of more game play for the user.

4. Project Background

The goal we hold for our final product is to create a game that is both entertaining and easy to play. The “naked nightmare” dream is one that we find to be easily relatable, and believe will be the perfect fit for the type of game we have set out to create. The idea allows for expansion with new features and added complexity. One of our team members has previous experience with SDL2 and will be able to provide the rest of us with a jumping off point. All of us have experience with C++, which is the language the game will be written in.

For a more detailed description of the project as a whole, please visit our Concept of Operations document.

5. Process

We are using an agile development model during this project. We decided on this based on the short amount of time in which we have to complete the game, and because the the models adaptability in the face of change and unexpected risks. Not all of us have experience with writing games nor SDL2, so this method will give us a chance to make adjustments as the project progresses according to what is wokring and what is not. Because code will be shared, this method assumes personal responsibility and consistant communication. This method will give team members rapid feedback that will facilitate quick problem solving and therefore productive development. In order to mazimize feedback we will have periodic group work sessions – along with more frequent stand-ups - for both regular updates and for when a team member is struggling with a problem that effects the work of the entire group. Working on problems such as these together will resolve them quickly and efficiently. The goal is to be able to adjust design as the project calls for, and to keep our team producing and on track for a successful finish.

6. Mechanisms, Methods, Techniques

For team members running a Mac environment, the XCode IDE will be used for development. We will be able to write and run the game in the IDE. Team Members running a Windows environment will be using Microsoft Visual Studio. Both of these IDE’s will be used in tandem with GitHub.

GitHub will be used to handle synchronizaiton and integration between team members. This will also be where our documentation is housed, so team members will be able to easily access any reference items they need. It will be used as our version control system, enabling us to roll-back changes that are unsuccessful or yield unpredictable results. This will also ensure that all of our work is never in danger of being lost, if a computer is either lost, broken, or stolen. Our work will also be accessible from anywhere with an internet connection if one of us needs to access their work from a different computer for any reason.

Development of this game will be done using the SDL2 development library. The user will be able to launch the game executable and play and pause the game.

During early brainstorming and development, the team drew from their own gaming experience and from additional research on common game development practices and themes.

7. Milestones

Successfully displaying main game window by Monday, May 9th

Art to be completed by Friday, May 13th

Collision detection present by Monday, May 16th

Line of Sight detection present by Monday, May 16th

Final Project to be completed on Friday, May 25th

Please refer to our meeting notes for more information about the process in actuality.

8. Resources and References

Piskel

Photoshop

Lazy Foo’ Productions

XCode

Microsoft Visual Studiosa

Meeting Notes

Meeting:

Discussion of necessary NPC’s that need to be designed,created, and animated.

Other art must also be created for level design and layout.

Will this take place indoors, outdoors, or both?

If indoor:

* Doors/Lockers
* Potted plants
* Walls/Windows
* Floor

If outdoor:

* Bushes/trees
* Grass
* Buildings
* Cars?

Also need to create the opening splash screen. Is there a possibility we could create an opening sequence of some kind? A short pre-made animated clip or something we could have run each time? The first could also be made just using text and still images which would cut down on the production time as that is not a mian goal of the project.

Would this clip play everytime it’s run or only the first?

Continued onto level sketching and talk of how to make the game a challenge without accidentally creating a game that is impossible.

A chunk of time here is spent discussing what angle of perspective to design the game from, and whethor it would be feasible to implement an isometric layout.

Meeting:

Discussion of art that we still need. We realize we will need different versions of the main nude character for each level if we decide to put a clothing token at the end of each level. On the subsequent level he will need to be wearing that item. Should be easy to just add color and create versions of the same character sprite.

Wall design still needs work, the angles are not coming together. Discussion and further sketching.

General talk about level design and testing.

Possibility of adding a timer as a feature. Could use to keep track of time limits on special “power-up” items. Whole range of what these could be – invisibility, speed boost… what if the invisibility was a cardboard box item that the character would pick up and put over himself! Could just show legs and feet moving the box – hilarious.

Timer could also be used to find high scores for fastest time completed. Having the timer show while game is running will add another aspect of pressure for the user

Meeting:

Discussion items to bring up: adding game states, timer, wiki, player detection.

Begun meeting with discussion of grading criteria as explained in class that day:

Need to add in developer logs now if they’re only in your local saves. Same for all documentation.

Include detailed instructions and easy way for Michal to play/test the game, i.e. test cases.

Game should be intuitive enough to just begin playing.

Make an extremely useful/concise README file

If something doesn’t fit there, README should point to it

Include:

Authors -> optional contact info if comfortable

Outline of what the project is

General list of how to build and program requirements

Basic navigation of code, so know how to navigate code for future addition of mods. etc.

Testing? (any kind of automated test opportunities)

Want to be able to read code easily -> be able to add things without much effort.

List of discussion of what needs to be/could be accomplished before deadline:

Art:

* Level Floors/Walls
* Clothes/Items
* Lockers/Plants
* Polished splash screen/pause screen

Text Class:

* # times seen
* clock
* powerup time
* text

Timer:

* Item time limits?
* Total game time -> save as game stats?

Detection:

* Line of sight
* Collision

Game States:

* Pause
* Menu
* Levels

Documentation:

* SRS?
* Use Cases
* Con-ops
* Architectural design diagram
  + slides for reference, q’s at end
  + Need comments that provide easy code navigation
* Installation Instructions
* Test Cases
* README updated to be useful and to guidelines
* Narrative of project?
* Developer Logs – continue updating
* Table of Contents? vs. sidebar…
* Story Boards

Concluded meeting with communal work time.

Meeting:

Begun with updates. Meeting generally dedicated to work time and discussion of items previously mentioned.

* [Home](https://github.com/MrCQuinn/Indecent-Exposure/wiki)
* [Project Concept](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Project-Concept)
* [Intended Use](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Intended-Use)
* [Concept of Operations](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Concept-of-Operations)
* [Risks and Risk Mitigation](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Risks-and-Risk-Mitigation)
* [Milestone Goals](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Milestone-Goals)
* [Initial Mockup](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Initial-Mockup)
* [Developer Logs](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Developer-Logs)
  + [Charlie Quinn](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Charlie-Quinn-Developer-Log)
  + [Portia Seater](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Portia-Seater-Developer-Log)
  + [Alex Brandenfels](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Alex-Brandenfels-Developer-Log)
* [Meeting Notes](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Meeting-Notes)
* [Team Source Force](https://github.com/MrCQuinn/Indecent-Exposure/wiki/Team-Source-Force)