**Sinister Transistor**

**Concept of Operations**

**COP 4331, Spring 2016**

Team Name: The Mega Bytes

Team Members:

* Greg Kelso
* Mark Boutwell
* Joel Gardyasz

Modification history:

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Who | Comment |
| v0.0 | 01/28/16 | Greg | Template |
| v1.0 | 02/3/16 | Greg | Operational Features & Implementation |
| v1.0 | 02/4/16 | Mark | Added Users and Modes of Operations |
| v1.1 | 02/4/16 | Joel | Current System & Proposed System Needs |
| V1.2 | 02/5/16 | Mark | Added Project/Team Name and Updated Title page |

Contents of this Document

The Current System

The Proposed System

* Needs
* Users and Modes of Operation
* Operational Scenarios
* Operational Features
* Implementation

**The Current System**

The idea for our project stems from 2D games like Legend of Zelda on the NES. We want to create a game that is closely related to titles such as this because our group was interested in game design and thought it would be a good fit for our software engineering project. We chose a 2D game instead of 3D one for simplicity and for getting those most out of what we can put into the game. If the game is easier to make then more features can be added. We would rather create an awesome 2D game than create a lackluster and seemingly rushed 3D game. There are many games like this one out at the moment; new games that are similar are being created almost daily. We simply want to put our own take on a game to learn more about the game design process and to learn what goes into projects like these.

**The Proposed System: Needs**

Creating a new game comes with a lot of decisions to be made regarding mechanics of the game, level design, character progression, etc. We will use what we’ve learned from playing other games to improve on these aspects and make our game the best it can be. Efforts will be made to make the game easy to pick up and play from the start, but also to keep the game fun and full of exciting things happening all the time.

**The Proposed System: Users and Modes of Operation**

Player: The player is using Windows, Linux, or OSX. The player will interact with the game using keyboard and mouse.

Modes of Operation:

Main Menu: The player will interact with the UI to change game settings, start the game, or exit the game.

Game: The player will use keyboard and mouse to control the player and navigate the levels.

**The Proposed System: Operational Scenarios**

For the standard scenario, the player will load up the game on Windows, Linux, or OSX and enter the main menu. The player will interact with the menu to start the game. When the game starts, the player will control a character on the screen to navigate a 2d environment and fight AI enemies. Once the player completes the level, the game will return to the main menu where the player may choose to exit the game.

**The Proposed System: Operational Features**

Must Have:

1. An environment for the player to move around in and interact with.
2. A player that the user controls, and contains multiple actions controlled with keyboard input.
3. Enemies for the player to fight.
4. A health system for both the player and enemies, in order to add engaging gameplay mechanics.
5. Objects for the player to interact with (could be currency to collect, or weapons to pick up and use).
6. Easily understandable explanation of the user controls, gameplay mechanics, and an objective for the player to work towards.

Would Like to Have:

1. An in-depth tutorial explaining controls, mechanics, etc. explained in #6 above.
2. A fully-fledged dungeon, similar to The Legend of Zelda on the NES, complete with increasing difficulty, power-ups, and a boss at the end.
3. Music and sound effects.

**The Proposed System: Implementation**

The video game will be developed using Unity2D. The content will be exported as a desktop standalone usable with Windows, OSX, and Linux. In addition to Unity's built-in tools, all scripting will be done in C#. The reason we chose Unity2D is because we all have some experience with the tool, and we believe it will be easier to get an impressive video game with the current time constraints by using 2D over 3D. We will be using C# because in our opinion it is better suited to handle the scale of a Unity project as opposed to Javascript.

The development will probably have a rough start as we get a feel for the technical requirements of the project, combined with the individual strengths and weaknesses we all bring to the table. Because of this, the only learning curve we will have to deal with is how we effectively communicate and work together on the same project, since we all have experience with Unity and C#. Our main limitation will be with version control. Unity offers version control for its paid users (which we aren’t), and from what we’ve heard, there tend to be a lot of merge conflicts when people use Git on Unity projects. We will be sticking with Git for now and hopefully not run into too many issues later in the project.

There are a lot of alternatives we could have considered instead of Unity2D. Unity3D was an option, but we believed it would be too much work for one semester. MonoGame, PyGame, and Unreal are also other game engines with a lot of documentation and support online. However once we realized that we all have experience with Unity, we decided to stick with what we knew. In addition to our personal experience with Unity, it is our opinion that it is better documented than other options at the moment, which will make developing the video game very easy.