**Dung Eon : A Dungeon Crawler**

**Concept of Operations**

**COP 4331, Spring, 2016**

Team Name: Bobby Sandwich

Team Members:

* Jacob Crandall
* Robert Bland
* Jacob Jiskoot

Modification history:

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Who | Comment |
| v1.0 | 02/03/16 | All Members | Initial Ideas for each topic in the table of contents |
| v1.1 | 2/05/16 | Crandall | Took out comments - Proofread |

Contents of this Document

The Current System

The Proposed System

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

**The Current System**

This project is essentially going to be a dungeon crawler, phone application. A dungeon crawler is a fantasy role-playing game. The user is the main character, the hero, and they attempt to navigate a maze in order to find their way to the next level, where the scenario repeats until the last level. There are different obstacles and enemies to defeat on each level.

Each level’s map is typically created randomly within certain parameters based which level it is. This would include where everything is: enemies, passage to next level, treasures, as well as potentially even which enemies and treasures show up at each level.

**The Proposed System: Needs**

The world revolves around entertainment and every person strives to gain some enjoyment out of their lives. So what we are proposing is a dungeon crawler video game application for Android purely for entertainment purposes. One of the main reasons why we are going to make this is to make an app that whenever anyone is bored or needs some time to themselves they can simply tap our app and start enjoying themselves again.

Another reason why people need this is that it will be completely free, unlike most other mobile apps out there. The reason for this is because we want everyone to enjoy our game regardless of of wealth, since some people feel like getting more and more mobile games that aren’t free will add up to a great expense.

Expanding on the free idea, our version of free isn’t the same as others, since some might think free means ‘freemium’, which for our case it is not. This is important because many other ‘free’ games often have lots of microtransactions, which is what we abhor. So what we are offering is a game that is completely free and accessible to anyone interested. This is different from many other games since they are typically free, but have so many paywalls and microtransactions that it makes the experience unenjoyable or can make the game feel grindy or tedious.

While our game will probably be very similar in many regards to other dungeon crawlers we hope that it can grow to be more than that. We hope to have a structure that is easy to upgrade and add functionality to. We will keep this project open source and hope that we or others may expand upon our initial work and that a community could grow out of the fact that the game is very accessible to modify.

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

We will only have 1 class of user.  
We will remain a free to play game with one difficulty setting

Therefore we only need to offer one standard mode of operation.

While we plan to target Android development first if resources allow we will move to iOS and therefore have to introduce new modes of operation then. One mode for Android and one for iOS but the modes, to the user, shall be the same.

**The Proposed System: Operational Scenarios**

This system will primarily be used for entertainment that involves the mind (strategy game), and testing our (this group’s) abilities to make a phone application. Our system will be a stand alone product, which will hopefully limit all of the errors to internal problems with the application and not problems with routers, or other phone applications. We will also look to limit the number of allowed inputs, meaning if the input is invalid it will never get processed initially so there won’t need to be any exception handling.

For instance, if the user clicks on a wrong spot on the map, nothing will happen because the input wasn’t accepted. Also if a user isn’t able to use a certain item for some reason, there won’t be any way for the user to use it aka no “use” or “equip” button, or maybe the button is grayed out, etc.

In the event of a system crash, since this is an entertainment game and not essential, we will probably close the system in the way that is the least taxing on the phone being used. Depending on the type of system crash we may also incorporate an autosave feature so the user can pick up where they left off. If the system crashed because of an in game scenario the user is looping through we will reset the user’s game to the closest previous save.

**The Proposed System: Operational Features**

Must Have:

Be able to run on Android

Main Menu - Play / Settings / etc.

A controllable character

Enemy + Connected AI

Usable items (Weapon, Defensive item)

Characteristics that all enemies + heroes have : Health, Attack, Defense

Enemies react on player moves ( Turn-Based )

Multiple randomly generated levels

UI with 1 inventory slot for each type of item

Enemies drop items

Would Like to Have:

Additional characteristics for enemies + heroes : Mana, Spells, Charisma, Intellect, Agility

Unlockable items / characters / achievements

Run on other mobile platforms

Multiple Character types

Multiple unique enemies

Increasing difficulty per level

Currency + Shop system

Leaderboard

In the android market

Multi - Inventory System

**The Proposed System: Implementation**

Initially we plan to release the game on Android.

However, a major goal of ours is for the game to be easily portable to other mobile platforms, as well as easily upgraded. For this reason we will be developing using Xamarin which provides a code "background" that all platforms may use. We will only have to update user interfaces and make slight adjustments depending on which platform the game is running on.

Once we had made the decision to use Xamarin it was an easy decision to also use a supporting library called "CocoSharp". CocoSharp is a 2d game engine that also has great support for easily developing a game on multiple platforms.

While these tools are phenomenal for our purposes we unfortunately have a huge learning curve.

Using Xamarin and CocoSharp locks us into the C# language. Unfortunately this means that we have a huge learning curve as none of our members have any experience in either C# or mobile development. The C# language and the tools available with it are however too good to pass up.

The fact that C# also supports the object-oriented paradigm is just another upside. We plan to use an object-oriented approach because it will allow us to give our game a familiar structure during the design phase. This is especially important to consider because better structure before going into the coding phase will allow us to more easily learn what we need to about

Xamarin and C# to complete our tasks.

It is difficult to know all the limitations of our environment before we begin working with the tools.

And that in itself is a major limitation of our system, our limited knowledge of the tools we will be using means there may be limitations we are not aware of yet. However one major limitation we do know of is that while Xamarin does make cross-platform development easier it will still introduce obstacles. Such as the fact that certain user interface elements will have to be coded separately on each platform. Another known issue is that using the Xamarin network will bloat our application in both file size and performance. However, these issues are true of similar tools we know of, and developing the game for each platform individually would go against our hope to make the game eventually portable. CocoSharp for our purposes shares pretty much every advantage/disadvantage that Xamarin does. The actual implementation itself should be pretty straightforward with the tools we are using. CocoSharp will let us do basic initialization of our game as well as defining the different steps to take depending on the platform. For this reason we will start with these two items. From there on splitting up our design into objects and eventually into code should be simple.

We will not require internet capabilities and there are few limitations of how our program should be used. There are also few ways that our game would pose as a security risk because it doesn not interact with anything outside of its system. After this, seeing as the purpose of the game is to entertain use should be left up to the user as much as possible.

Although this may be a side-note a major goal of this project is to be a learning experience for our developers. For this reason we may implement many different features rather than developing features in great detail. The goal is that the depth of these features may be developed outside of this project or in different stages at will. For example once we have implemented a weapon system we may spend time developing a unique map generation algorithm, or a path finding algorithm rather than creating hundreds of unique items. The goal is to put into place a system that makes upgrades such as that easier. So at first in implementation we hope to create a transparent framework to make future upgrades simple. It is our hope that our tools, environment, language and ideas reflect and aid these goals.