**Dung Eon : A Dungeon Crawler**

**Concept of Operations**

**COP 4331, Spring, 2016**

Team Name: Bobby Sandwich

Team Members:

* Robert Bland
* Jacob Crandall
* Jacob Jiskoot

Modification history:

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Who | Comment |
| v1.0 | 2/17/2016 | Jacob Jiskoot | Initial Notes |
| v1.1 | 2/21/2016 | Jacob Crandall | Added a little to initial notes & Added Use Case section |
| v1.2 | 2/25-26/2016 | All Team | Team completed sections 3 and 4 |
| ... |  |  |  |

Contents of this Document

Introduction

·         Software to be Produced

·         Reference Documents

·         Applicable Standards

    Definition, Acronyms, and Abbreviations

Product Overview

·         Assumptions

·         Stakeholders

·         Event Table

·         Use Case Diagram

·         Use Case Descriptions

Specific Requirements

·         Functional Requirements

·         Interface Requirements

·         Physical Environment Requirements

·         Users and Human Factors Requirements

·         Documentation Requirements

·         Data Requirements

·         Resource Requirements

·         Security Requirements

·         Quality Assurance Requirements

Supporting Material

**Section 1: Introduction**

Software to be Produced:

* This project intends to create a dungeon crawler phone application.  This software will be a standalone product and is the only necessity for our system.  The bulk of the application is being created with the potential to run on multiple phone operating systems.  More depth into the scope of this project can be found in the Concept of Operations document or at the start of the Project Management Plan.  Check out any of the following reference documents to get a better idea of the proposed system.

Reference Documents:

* Concept of Operations
* Project Plan
* Support Docs folder (although most are included as images here)

Applicable Standards

* Do Factory’s C# Standards : <http://www.dofactory.com/reference/csharp-coding-standards>

Definitions, Acronyms, and Abbreviations

* Dungeon Crawler - 2d maze game with enemies and usable items
* CoCos Sharp - Additional outside code that acts as a game engine

**Section 2: Product Overview**

Assumptions:

* We are assuming the user of our application has a basic working android smartphone (as it is sold) with a working touch screen.
* We will assume that the user will have enough room on their phone for us to be able to push updates OR that their operating system will handle this for us.
* We will assume that the phone will assist in the quitting of the application.  So that when we exit the application the android will handle or help with this process.
* We will assume during run-time that android OS will protect our memory from other applications, as well as ensuring that we don’t accidentally interfere with other processes.  
    
  Stakeholders:
* Users (should we break users up into different types aka hard core dungeon crawler? vs average individual?)  
  Users will want a game with lots of content, interesting gameplay, easy to use and that runs with little to no performance issues.
* Other developers that may wish to view or add to our code  
  Developers will wish that our code is both organized and documented well.  They will also hope that our code is easily modifiable and easy to add to.

Event Table:

|  |  |  |  |
| --- | --- | --- | --- |
| Event Name | External Stimuli | External Responses | Internal data and state |
| Map Touch | Touch open map space | Hero move to spot | A level is loaded and user is not in a menu |
| Map Item Touch | Touch item on map | Pick up item - put in inventory | The playable character is on an item - attach item class to user class |
| Equip Weapon | Touch weapon in bag / menu | Equip weapon sprite | On a bag/menu and there is an equippable item - load weapon class, de-load old weapon class |
| Use Item | Touch item in bag | Execute specific use item’s effect - if it has visuals | On a bag/menu and there is a usable item - remove item class from user class - apply use item’s effect |
| Fight Enemy | Touch enemy | Move towards the enemy and attack it | There is an attackable enemy available - call damage functions for enemy |
| Enemy Hit | None | Show hit | If you are in a fight with an enemy and they land a hit on you : damage character : Update health UI |
| Go To Next Level | Touch next level portal | Move to portal and load next level | The current level is cleared and a new level is loaded |
| Play Button | Touch play button | Load a new level | On the main menu hit the play button - will create a game instance including all related objects |
| Quit Button | Touch quit button | Exit the application, go to android homescreen | Call quit function |

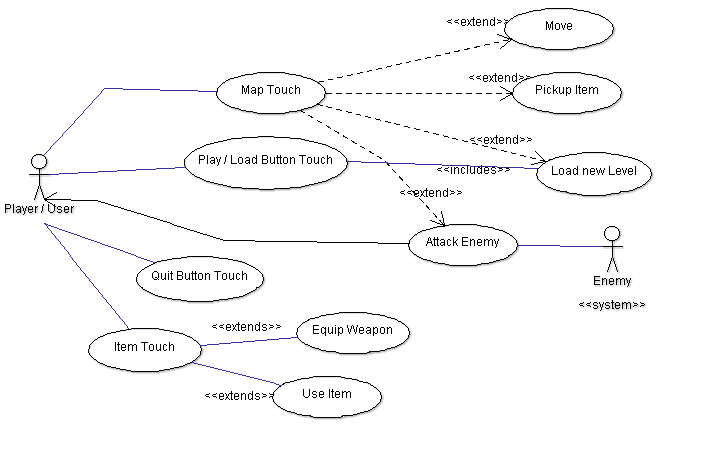


Figure 2.1 Use Case Diagram

Use Case Descriptions:

From top to bottom

* Map Touch - Checks that a valid tile has been touched.  If a valid tile was touched then do additional checks for each item.  Each extends is an additional extra item if a checks pass.
  + Move - A valid tile was touched and there is a valid path to that tile
  + Pickup Item - There is an item at a valid tile location.
  + Load New Level - A tile that represents the end of the level was touched
  + Attack Enemy - A tile containing an enemy was touched - call functions to attack that enemy
* Play / Load Button Touch - Initialize a new game - load a new level
* Quit Button Touch - Exit the game - assume that android / xamarin / CocosSharp handles garbage collection
* Item Touch - Upon touching an item on the UI check if it is an equippable item or a usable item
  + If it is a usable item use it
  + If it is an equippable item equip it
* Enemy attacks you - Attack a user character if they are within attacking distance

**Section 3: Specific Requirements**

TEMPLATE:

|  |
| --- |
| No: <unique requirement number> |
| Statement: <the "shall" statement of the requirement> |
| Source: <source of the requirement> |
| Dependency: <list each other requirement on which satisfaction of this requirement depends. (May be "None")> |
| Conflicts: <list each other requirements with which this requirement conflicts. (May be "None")> |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method: <How can you tell if the completed system satisfies this requirement? > |
| Revision History: <who, when, what> |

3.1 Functional Requirements

|  |
| --- |
| No: MenuReq |
| Statement: The software shall have a Main menu when starts up with multiple buttons that are pressable for the user |
| Source: Storyboard, Figure 2.1 |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: Storyboard, Figure 2.1 |
| Evaluation Method: When the software starts it should show a main menu and try out each button |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

|  |
| --- |
| No: MapReq |
| Statement: Player shall do one of the four options when pressing on the map:   1. Move to target location 2. Move to target location and pick up item 3. Move to load new level 4. Possibly move and attack the enemy |
| Source: Storyboard, Figure 2.1 |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: Storyboard, Figure 2.1 |
| Evaluation Method: Test each individual case |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

|  |
| --- |
| No: ItemReq |
| Statement: When an item is moved onto the player is given two choices   1. Equip Item 2. Use item |
| Source: Storyboard, Figure 2.1 |
| Dependency: MapReq |
| Conflicts: INTR2, INTR5 |
| Supporting Materials: Storyboard, Figure 2.1 |
| Evaluation Method: Moving over equipment and trying to both use and equip it |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

|  |
| --- |
| No: PlayReq |
| Statement: The Main Menu shall have a play/load button to press that will start the game |
| Source: Storyboard, Figure 2.1 |
| Dependency: MapReq |
| Conflicts: MapReq |
| Supporting Materials: Storyboard, Figure 2.1 |
| Evaluation Method: By pressing the button and trying to play/load the level successfully |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

|  |
| --- |
| No: ControlReq |
| Statement: Game shall be controlled by user pressing on the phone |
| Source: Storyboard, Figure 2.1 |
| Dependency: INTR3 |
| Conflicts: INTR3 |
| Supporting Materials: Storyboard, Figure 2.1 |
| Evaluation Method: Storyboard, Figure 2.1 |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

|  |
| --- |
| No: CreditsReq |
| Statement: The main menu shall have a credits button |
| Source: Group Discussion |
| Dependency: MainReq |
| Conflicts: MainReq |
| Supporting Materials: Group Discussion |
| Evaluation Method: Pressing the Credits button from the main menu |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

|  |
| --- |
| No: SoundReq |
| Statement: The software shall have music and sound effects |
| Source: Group Discussion |
| Dependency: INTR1-6 |
| Conflicts: None |
| Supporting Materials: Group Discussion |
| Evaluation Method: Go into the game and go to multiple areas that have different music and test all objects that are associated with sounds and music in game. |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

3.2 Interface Requirements

|  |
| --- |
| No: INTR1 |
| Statement: The level tiles shall provide the ability to check which objects it contains. |
| Source: Figure 2.1 |
| Dependency: None |
| Conflicts: None |
| Supporting Materials:Figure 2.1 |
| Evaluation Method: Testing each individual case of touch. |
| Revision History: Jacob Crandall - Initial commit - 2/26/2016 |

|  |
| --- |
| No: INTR2 |
| Statement: The level tiles shall provide the ability to contain other characters and items and display the items contained within it. |
| Source: Story board / Figure 2.1 |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: Story board / Figure 2.1 |
| Evaluation Method: If a tile can easily add or remove objects from it, and all types of objects. |
| Revision History: Jacob Crandall - Initial commit - 2/26/2016 |

|  |
| --- |
| No: INTR3 |
| Statement: Characters (enemies / user) shall provide methods that interact with each of their stats (namely health). |
| Source: Figure 2.1 |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: Figure 2.1, UML Class diagram |
| Evaluation Method: If each user stat can be affected by another object |
| Revision History: Jacob Crandall - Initial Commit - 2/26/2016 |

|  |
| --- |
| No: INTR4 |
| Statement: All objects listed within the section 4 UML class diagram shall provide a “CCSprite” type object for interaction with CocosSharp |
| Source: UML Diagram / We need visuals and this is how CocoSharp interacts with them |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: UML Diagram |
| Evaluation Method: If CocoSharp recognizes our visuals |
| Revision History: Jacob Crandall - Initial Commit 2/26/2016 |

|  |
| --- |
| No: INTR5 |
| Statement: Character classes shall provide the functionality to use item objects |
| Source: UML Class Diagram / Figure 2.1 |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: UML Class DIagram / Figure 2.1 |
| Evaluation Method: If a user can use both equippables and use items. |
| Revision History: Jacob Crandall - Initial Commit - 2/26/2016 |

|  |
| --- |
| No: INTR6 |
| Statement: Enemy type characters shall be able to detect when they are near a user type character and engage combat. |
| Source: Figure 2.1, Storyboard |
| Dependency: INTR3 |
| Conflicts: None |
| Supporting Materials: Figure 2.1 / Storyboard |
| Evaluation Method: If a user can attack an enemy |
| Revision History: Jacob Crandall - Initial Commit - 2/26/2016 |

3.3 Physical Environment Requirements

|  |
| --- |
| No: AOS |
| Statement: Software shall run on a phone with an Android OS |
| Source: Project management plan |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: None |
| Evaluation Method: When the program starts successfully on an Android phone |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

3.4 User and Human Factors Requirements

|  |
| --- |
| No: HF1 |
| Statement: The system shall detect all correct input and have simply NO action on an incorrect touch / input.  For example, trying to go to a map tile the user character cannot access will have NO effect. |
| Source: Storyboard |
| Dependency: INTR2 |
| Conflicts: None |
| Supporting Materials: Storyboard |
| Evaluation Method: |
| Revision History: Jacob Crandall - 2/26/2016 - Initial commit |

|  |
| --- |
| No: HF2 |
| Statement: The system shall be intuitive.  We hope that there is no need for tutorials for any type of user. |
| Source: Storyboard |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: Story board |
| Evaluation Method: Provide the system to a varied audience and see if they can figure out each component of the system. |
| Revision History: Jacob Crandall - 2/26/2016 - Initial commit |

3.5 Documentation Requirements

|  |
| --- |
| No: OnlineDoc |
| Statement: The documentation shall be online on github available to everyone on release |
| Source: Group Discussion |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: N/A |
| Evaluation Method: Trying any of the documentation listed here  https://github.com/jacobmcrandall/POOPCrawler/blob/master/ProjectManagementPlan.docx |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

3.6 Data Requirements

|  |
| --- |
| No: DARLVL |
| Statement: Randomly create the level with a winnable path, enemies, and usable items |
| Source: Project Management Plan |
| Dependency: Functional, without this there is no game. |
| Conflicts: None |
| Supporting Materials: N/A |
| Evaluation Method: Every map the user encounters will be winnable.  There aren’t usable items and enemies on every tile. |
| Revision History: Jacob Jiskoot- Initial commit - 2/25/2016 |

|  |
| --- |
| No: DARDIJK |
| Statement: Use Dijkstra’s algorithm to move the character most efficiently |
| Source: Group Discussion |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: N/A |
| Evaluation Method: The character chooses the shortest path to a spot |
| Revision History: Jacob Jiskoot- Initial commit - 2/25/2016 |

|  |
| --- |
| No: DARDAM |
| Statement: Calculate damage done to player and enemies while fighting |
| Source: Group Discussion |
| Dependency: INTR3 |
| Conflicts: DARHEAL |
| Supporting Materials: N/A |
| Evaluation Method: Enemy and player take appropriate damage in fights |
| Revision History: Jacob Jiskoot- Initial commit - 2/25/2016 |

|  |
| --- |
| No: DARHEAL |
| Statement: Player will heal overtime |
| Source: Group Discussion |
| Dependency: None |
| Conflicts: DARDAM |
| Supporting Materials: N/A |
| Evaluation Method: User heals certain amount over specified time |
| Revision History: Jacob Jiskoot- Initial commit - 2/25/2016 |

* Not saving anything so no data is retained.

3.7 Resource Requirements

|  |
| --- |
| No: XamCoReq |
| Statement: The game shall be made with Xamarin and CoCos Sharp |
| Source: Project Management Plan |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: N/A |
| Evaluation Method: Try to compile and run game to see if it runs |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

|  |
| --- |
| No: FundsReq |
| Statement: The funds shall be twenty dollars for buying sprites |
| Source: Group discussion |
| Dependency: Graphics requirement |
| Conflicts: None |
| Supporting Materials: N/A |
| Evaluation Method: Checking the balance at any time for the project |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

|  |
| --- |
| No: SpaceReq |
| Statement: The software shall not take up more than 10 MB of space |
| Source: Group Discussion |
| Dependency: Images and Code size |
| Conflicts: None |
| Supporting Materials: N/A |
| Evaluation Method: Checking file size when the game launches |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

|  |
| --- |
| No: ProcessReq |
| Statement: The software shall not exceed 70 % of CPU processor |
| Source: Group Discussion |
| Dependency: Efficiency of software |
| Conflicts: None |
| Supporting Materials: N/A |
| Evaluation Method: Check how much of the CPU is being allocated to the software when using it on Andriod |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

3.8 Security Requirements

* N/A Our application doesn’t deal with any sensitive information nor does it have any special access to anything important in the phone

3.9 Quality Assurance Requirement

|  |
| --- |
| No: TestReq |
| Statement: Software shall be tested after every increment added to the software |
| Source: Group Discussion |
| Dependency: This depends on every requirement since it checks every requirements |
| Conflicts: None |
| Supporting Materials: Figure 2.1 |
| Evaluation Method: By testing this after every increment to the game |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

|  |
| --- |
| No: BugReq |
| Statement: When an error occurs the software shall notify the user of the general class of bug |
| Source: Group Discussion |
| Dependency: All data requirements |
| Conflicts: None |
| Supporting Materials: Figure 2.1 |
| Evaluation Method: Force an error to see if notification pops up |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

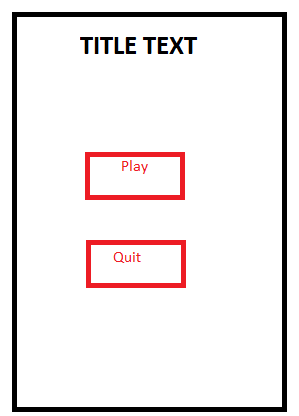
|  |
| --- |
| No: RelReq |
| Statement: The software shall have at most three bugs per month |
| Source: Group Discussion |
| Dependency: All Data Requirements |
| Conflicts: None |
| Supporting Materials: Figure 2.1 |
| Evaluation Method: By running the program at least 5 times a week and playing it for one month to see how many errors occur |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

|  |
| --- |
| No: AvailReq |
| Statement: The program shall be free for customers to use |
| Source: Group Discussion |
| Dependency: None |
| Conflicts: None |
| Supporting Materials: N/A |
| Evaluation Method: When released we will try to download it from the Android app store if possible, or host it somewhere else if necessary |
| Revision History: Robert Bland -- 2/26/16 -- Creating Requirements |

**Section 4: Supporting Material**

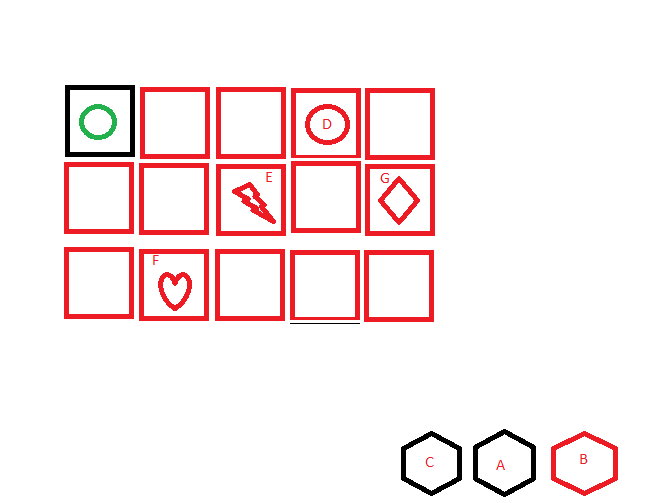
Story Board : This is a VERY quick rough mockup of what a user could interact with in given situations.

RED ITEMS : Interactable with a user



-Play - Loads game

-Quit - Exits application to android home screen



A - Offensive item slot

B - Usable item slot - Clicking uses item

C - Defensive item slot

D - Enemy - Clicking moves NEXT to enemy and begins combat

E - Equippable item - Clicking moves to the item, drops the current equippable item and replaces it with that one

F - Usable item - Clicking moves to the item, drops the current use item (if applicable) and replaces it with that one

G - End level portal - Ends this level and loads a new one

All other red tiles are valid map tiles to move to and upon clicking will move the user (green circle) here.

Class Diagram : Note this is a sketch up from ideas and is subject to change

