**Software Requirements Specification**

Dungeon Adventure 2.0

**Prepared by Darence Thong, Jennifer Boutell, JP Montagnet**

**UW Tacoma GC-SDE**

**1/30/22**

**Table of Contents**

**Table of Contents**

**Revision History 1**

**1. Introduction 2**

1.1 Purpose 2

1.2 Document Conventions 2

1.3 Intended Audience and Reading Suggestions 2

1.4 Project Scope 2

1.5 References 2

**2. Overall Description 3**

2.1 Product Perspective 3

2.2 Product Features 3

2.3 User Classes and Characteristics 3

2.4 Operating Environment 4

2.5 Design and Implementation Constraints 4

2.6 User Documentation 4

2.7 Assumptions and Dependencies 4

**3. System Features 4**

3.1 Maze 4

3.2 Hero 5

3.2 Enemies 5

3.2 Game State Storage 6

**4. External Interface Requirements 7**

4.1 User Interfaces 7

4.2 Hardware Interfaces 7

4.3 Software Interfaces 7

4.4 Communications Interfaces 7

**5. Other Nonfunctional Requirements 7**

5.1 Performance Requirements 7

5.2 Safety Requirements 7

5.3 Security Requirements 8

5.4 Software Quality Attributes 8

**6. Other Requirements 8**

**Appendix A: Glossary 8**

**Appendix B: Analysis Models 9**

**Appendix C: Issues List 10**

# Revision History

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Team | 1/30/22 | Initial draft | 1 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 

# **Introduction**

## **Purpose**

Dungeon Adventure is a fantasy role playing game, in which an adventurer navigates through a maze, experiences challenges, and collects items. The goal of the game is to identify the exit while in possession of the required items. Dungeon Adventure 1.0 was released in December, 2021.

This document provides a detailed description of Dungeon Adventure 2.0 for developers and stakeholders. It will describe technologies used, design and features, software and hardware requirements, interface/GUI, software constraints, and the scope of this project.

Additionally, this document will aid the software developer team in the construction of the second version of this dungeon project.

## **Document Conventions**

Subject headings are indicated by Times New Roman, bold font, and numbered by section and subsection. Subject content is provided in Ariel font, normal typesetting.

## **Intended Audience and Reading Suggestions**

Software developer engineers and students are the primary targeted audience. Related stakeholders can also find utility from this document.

The suggested sequence for reading this document for developers and project managers is to begin with the overview sections and proceed through all other sections in order. The suggested sequence for other stakeholders is to read through the introductory sections and select other sections as relevant to the readers’ purpose, from the table of contents.

## **Project Scope**

The purpose of Dungeon Adventure 2.0 is to provide a fun gaming experience for the user. This project will allow creators to refactor code that is well tested and exemplifies object oriented programing, add additional features that will utilize a SQLite database, add GUI, and create a more varied and interesting gaming experience than version 1.0.

## **References**

Dungeon Adventure 1.0 assignment from TCSS502, Fall 2021.

# **Overall Description**

## **Product Perspective**

This project is a second version. The first version was completed in December 2021. The intent of this version is to add additional features, code refactoring, and add GUI.

Features that will be added are multiple hero characters to pick from and introduction of enemy characters. Enemies will have hit points and varying attack abilities that the hero character will encounter.

The previous version that requires the player to find 4 pillars to win while encountering health potion, vision potion, and pits will be preserved.

There will be multiple levels for the hero to traverse, such as a 3-story castle. The hero must find the staircase at each level to advance to the next. A “boss” enemy must be defeated before unlocking the staircase.

An updated UML will outline the entire program, showing the class relationships in object oriented programming.

Finally, a video demonstration will be recorded to show how a user would play the game.

## **Product Features**

Enemy characters will be stored in a SQLite database. Enemies are generated into this database and then retrieved during the generation of the dungeon.

The GUI will be created using TKinter. Sound effects and music will be incorporated with Tkinter as well.

## **User Classes and Characteristics**

**User**: Game user is proficient in English language at a minimum 5th grade reading level, and familiar with use of either a keyboard and pointing device. Unlikely to explore features when not explicitly required.

**Fanatic**: Possesses characteristics of User class, plus advanced understanding of the fantasy genre, RPG gameplay and tolerance for terrible jokes. Likely to explore most or all features of the game.

**Technical:** Typically a tester or instructor. Expected to traverse the game more extensively than other users and identify potential technical issues. May potentially overlap with the Fanatic class.

## **Operating Environment**

This software can be operated in Windows, Mac, and Linux operating systems, with Python 3.x installed. The user must have access to a command prompt in order to initialize the game.

## **Design and Implementation Constraints**

A computer with the current version of any Linux, Windows, or Mac based operating system should be sufficient to run Dungeon Adventure 2.0

## **User Documentation**

User documentation will be provided in-game with use of GUI. Guidelines on how to play the game will be explained.

More specific instructions to be added here.

## **Assumptions and Dependencies**

Dependencies include TKinter and Pytest for development of this project.

# **System Features**

## Maze

The game shall generate a maze for transversal by the player.

### Description and Priority

Maze is generated with a randomized pathway for the player. Essential objects and exit are placed after the maze pathway is created. There is built-in validation of playable pathways with obtainable required objects. In version 2.0, improvements will be made in the programming code on clarity, reliability and in testing.

This is an essential feature.

### Stimulus/Response Sequences

MSR-1: Maze is auto generated after hero selection

MSR-2: Built-in maze validation automatically runs to ensure maze is playable including the ability to obtain required items to win. If a maze is determined to be not valid, another maze is generated until a valid maze is determined.

MSR-3: A new randomized maze is generated if the player loses or wins and desires another challenge.

## Heroes

A new set of diverse heroes are introduced. Each character class has weaknesses and strengths.

### Description and Priority

The Player will choose from 3 different heros:

**H-1: Warrior** – has a Crushing Blow attack which does significant damage to enemies.

**H-2: Priestess** – has an ability to heal.

**H-3: Thief** – can perform a surprise attack allowing an extra attack.

Each of the different hero classes have different attack strength and speed values, as well as health points.

This is an essential feature.

### Stimulus/Response Sequences

HSR-1: Player selects 1 of 3 hero classes at the beginning of the game.

HSR-2: Upon entering a Dungeon room containing an Enemy, the Player's hero interacts with that Enemy. Player’s hero may choose to fight with the Enemy, and may continue doing so, until either character’s health reaches 0.

## Enemies

### Description and Priority

The Player can encounter and fight 3 different enemies, each with weakness and strengths:

**E-1: Ogre** – has a shield that makes it difficult for warriors to defeat.

**E-2: Gremlin** – able to move very fast, rendering thieves’ surprise attack useless.

**E-3: Skeleton** – has a “silence” ability that nullifies healing magics.

The term “Monster” may be used interchangeably with “Enemy”, but the latter term recognizes that some Enemies potentially may not be Monsters (except in the metaphorical sense).

Each of the above Enemy types have different attack strength and speed values, as well as amount of health points.

This is a high priority feature.

### Stimulus/Response Sequences

ESR-1: Enemies are randomly placed in dungeons during the generation of the game.

ESR-2: Enemies in a dungeon room interact with Hero, if present. enemy persists until either its health reaches 0, at which point it dies.

ESR-3: Enemy strengths and weaknesses are dependent on the Hero class that interacts with the Enemy type.

## Game State Storage

It should be possible to interrupt game play, save game state, quit the application, and then resume gameplay at a later time.

### Description and Priority

Game state includes, but may not be limited to: enemy characters, player state, inventory, game play tuning factors, and map features. During restore of a saved game, these objects will be retrieved from the database and reinstantiated.

This is a high priority feature.

### Stimulus/Response Sequences

This feature is auto-generated at the beginning of the game. Save and load is triggered by the player which additionally triggers changes in the database.

### Functional Requirements

TBD

# **External Interface Requirements**

## **User Interfaces**

Tkinter will be used to develop GUI. This will allow the user to move through a maze by clicking or keyboard entry of moving up, down, left, right. Users will also be able to see the map of traversed maze, encounter of enemy characters, and user’s hero character information.

## **Hardware Interfaces**

Users input is via a mouse and keyboard and output, or touch-based equivalents. Output through a full-sized graphical display at least 11 inches diagonal. It is not intended for use on smaller mobile devices.

## **Software Interfaces**

This is a self-contained application and does not interface with software other than its host operating system.

## **Communications Interfaces**

No communication protocol is required for this single-player, offline game.

# **Other Nonfunctional Requirements**

## **Performance Requirements**

Performance requirements are minimal for this single player lightweight game. The visual display should update within one second of any user action.

## **Safety Requirements**

The user may encounter bad jokes. Although not required, access to anti-nausea medication is recommended as a preventive measure.

## **Security Requirements**

There are no security requirements.

## **Software Quality Attributes**

**Interoperability-1:** The game should behave roughly the same on all supported operating systems.

**Testability-1:** A decoupled approach in programming should allow unit-testing of all game internals. Automated unit-testing of graphical features is explicitly excluded.

**Reliability-1**: The game maze generation shall guarantee that any maze presented to the user can be completed.

**Robustness-1**: The game should not crash. Game termination should only be by completion of the game via win or loss, or saving of game state for later resumption.

# **Other Requirements**

To be determined. Left in place for later drafts.

**Appendix A: Glossary**

**Dungeon** - Representation of a physical space consisting of a Maze and all contents therein. The Maze itself is not dangerous to the Hero; its contents often are.

**Enemy** - Opposing non-player characters that the Hero encounters and must defeat.

**Entrance** - Starting point for the Hero in the Maze. Abuts the perimeter of the Maze.

**Exit** - It’s your freedom from the Maze. Find this location to escape but only after you have found all 4 pillars. Abuts the perimeter of the Maze.

**Health points** - Bounded set of units representing a character’s vitality. When a character’s Health points are reduced to zero, the character “dies”.

**Health potion** - Type of Potion that restores some portion of lost Hero health points.

**Hero** - Character that represents a user playing the game, traversing a dungeon maze. The character has health points that the user must maintain greater than zero or fails in the game.

**Load game** - Resume a saved game and continue where you left off, before you were so rudely interrupted.

**Maze** - A set of interconnected passages, with a single entrance and a single exit. Hero traverses this as part of gameplay.

**Monster** - Roughly synonymous with Enemy, but typically limited to beasts, phantasms and politicians.

**Pillar** - One of four pillars that the player must collect before exiting the dungeon to win the game.

**Pit** - Stationary peril that causes Hero to fall and lose health points.

**Potion** - Object that Hero picks up. It can be a Health potion or Vision potion.

**Save game** - Save your current location and current progress in the game to resume later.

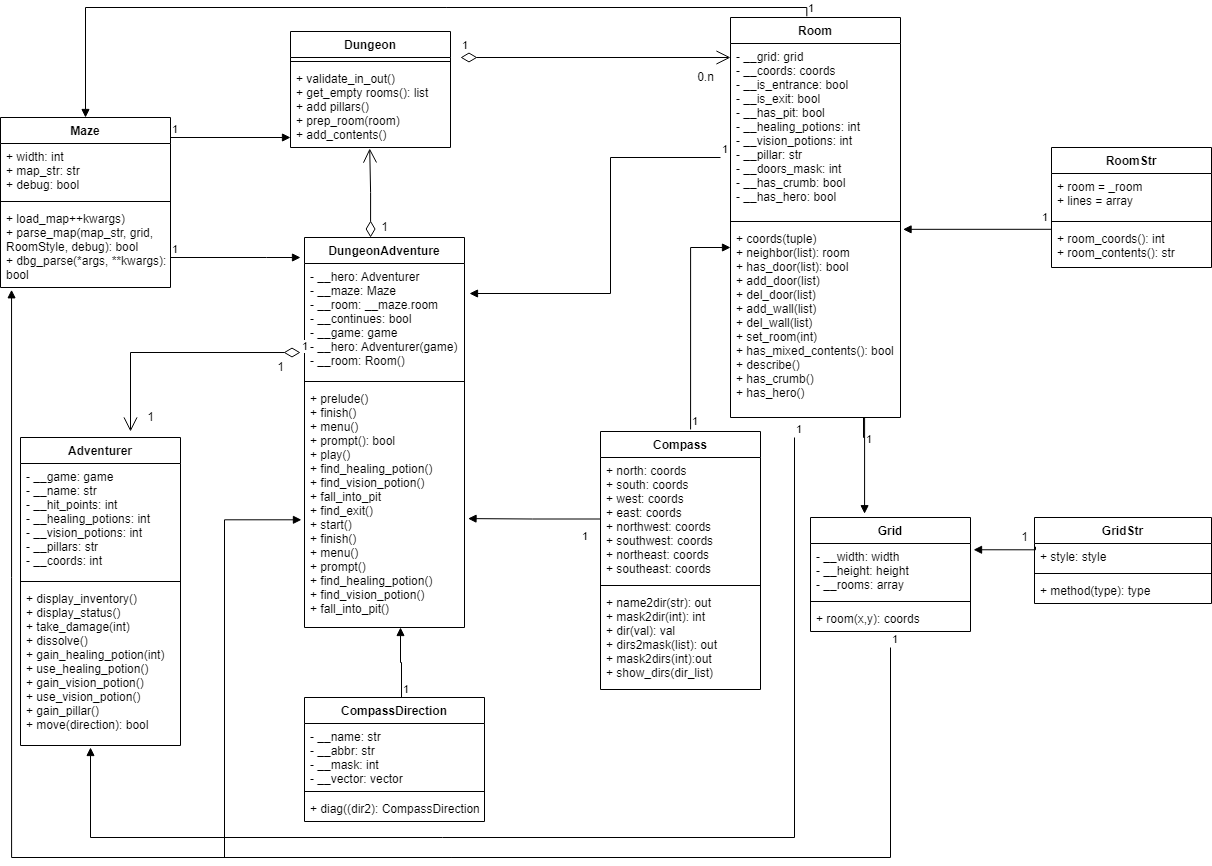
**SRS** - Software Requirement Specification.

**Vision potion** - Type of Potion that reveals nearby rooms. This lets the user briefly see the 8 rooms surrounding the current room.

**Appendix B: Analysis Models**

[Screenshot of UML 2.0 here]

**UML from dungeon adventure 1.0**



**Appendix C: Issues List**

Possible additions:

1. Sound/music
2. Bomb potion
3. Bosses at each level
4. Multi-hero party