Console-based Map Exploration Game

Software Requirements
Specification

Draft 1

February 17th, 2020

Greg Dagnan

Trevor Lee

Christopher Morales

Maliha Wahab

`

1 Introduction

1.1 Purpose

The purpose of this document is to establish our vision for a console based exploration and puzzle solving game of our own source material dealing with a post-apocalyptic world.

1.2 Scope

This game will consist of a MVC pattern with: text-based console output (JAVAFX GUI time permitting), game logic written in JAVA, and database persistence using SQL lite.

1.3 Definitions, Acronyms, and Abbreviations

JVM: Java Virtual Machine.

dB: database

1.4 References

Assignment specs provided by class administer.

1.5 Overview

The remainder of this document will consist of a description of the game as well as the requirements we have regarding its implementation.

2 Overall Description

2.1 Product Perspective

With a renewed interest in retro games, we feel that a return to the old console-based text puzzle games is ripe for reexamination for our post-apocalyptic adventure. This will be a stand-alone game-engine that should be designed to add additional chapters as they are released. Additional game details have been included in Product Requirements Detail Document.

2.1.1 System Interfaces

This game should be designed with a text interface (console is sufficient) and game engine written in Java version 8 or greater and utilizing a SQL Lite database.

2.1.2 User Interfaces

The user interface that is expected is a simple text-console that accepts text commands from the user and displays the results of the user's commands. If time-permitting, some of this functionality could be transferred to a GUI, but that is not a hard requirement at this stage.

2.1.3 Hardware Interfaces

The hardware requirements for this game are truthfully monitor, keyboard and a system with the JVM and SQL lite installed.

2.1.4 Software Interfaces

The game will interface with SQL Lite for persistence purposes but has no requirement at this time for additional software.

2.1.5 Communications Interfaces

NA

2.1.6 Memory Constraints

With the established above requirements, the memory requirements for this game should be minimal, but there must be awareness of any potential memory leaks that could occur while interfacing with the database

2.1.7 Operations

Normal operations for the user will consist of:

- *Entering commands into the console*
- *Saving game progress.*
- *Signing into game account.*

2.2 Product Functions

The game software will provide a user interface for user commands, utilize control logic to validate those commands, and persist game progress utilizing the SQL Lite dB for each user game account.

2.3 User Characteristics

This game can be enjoyed by anyone age 13 and above. Age requirement will help with comprehension and is necessary as there are graphic images described that might be unsuitable for children.

2.4 Constraints

There are no foreseeable constraints on this software other than the need for the JVM and SQL Lite preexisting on the system.

2.5 Assumptions and Dependencies

If this game begins to factor in GUI there might be additional requirements depending on the system put in place. However there are no foreseeable dependencies at this juncture..

2.6 Apportioning of Requirements

NA

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The game will employ a Java text console that parses commands from typed user input and check for validity. The console will prompt for input after each typed command and should display any results as text on the console.

3.1.2 Hardware Interfaces

User will interface with game via keyboard to enter commands into UI to get results out of game engine onto monitor. The simplicity of the game should mean that any modern system can play it. Dependencies are JVM and SQL Lite.

3.1.3 Software Interfaces

NA. Should not interface with additional software other than dB business requirement. Product should be designed with MVC design pattern to facilitate future enhancements.

3.1.4 Communications Interfaces

NA. If further developed, internet connectivity might be a necessary feature, but is unnecessary at this time.

3.2 Software Product Features (Also see attached Product Requirements Detail document)

3.2.1 Feature 1

Purpose:
To establish consistent gameplay for user.

Associated Functional Requirements
Functional Requirement 1: Text Based Commands (see Product Requirements Detail Document for list of commands)

- *Accepts Text Based Commands*
- *Validates input*
- *Directs command to proper sub-section of the game engine*
- *Returns results*

Functional Requirement 1a: User enters Incorrect Command:

- *Incorrect command entered.*
- *Message returned stating command is incorrect.*

Functional Requirement 2: Saves Progress to dB

- *User can tell system to save progress*
- *Game engine writes necessary information to dB.*
- *Verification is returned to Game Engine*
- *Game Engine informs player that game is saved.*

Functional Requirement 2a: Save process doesn't complete

- *User tells game to save*
- *Game attempts save but encounters error*
- *Message returned stating save did not complete.*

Functional Requirement 3: Game has hint system

- *Player asks for hint at console*
- *Game engine determines current room and items*
- *Game Engine interfaces with dB for available hints.*
- *Hints are returned and to engine.*
- *Game engine sends hint information to UI for display*

Functional Requirement 4: Game displays usable commands

- *Player asks for usable commands at console*
- *Game engine determines current room and items*
- *Game Engine interfaces with dB for available commands.*
- *Commands are returned to engine.*
- *Game engine sends available commands to UI for display*

Functional Requirement 5: Game has item system

- *Player initiates a new game.*
- *Game engine interfaces with dB for information on items, both fixed and dynamically located*
- *dB returns item information.*
- *Game engine converts information into necessary item object placement and renders them accordingly*
- *UI receives information on item availability as user interacts and user stores items as desired.*

Functional Requirement 5: Game has combat system

- *Player initiates a new game.*
- *Game engine interfaces with dB for information on enemies, both fixed and dynamically located*
- *dB returns enemy information.*
- *Game engine converts information into necessary enemy placement and stats and renders them accordingly*
- *UI receives information on enemy encounters as user encounters them on map.*

3.3 Performance Requirements

Static: the system only requires a single keyboard and monitor. There is only one execution of game on a system at any time (not multiuser).

Dynamic: amount of transaction will depend on game circumstances and command, but they should be minimal. As the design is simple, no transaction should last more than 5 seconds. Since there is no multithreading required, there should be a straightforward single transaction system on any given command.

3.4 Design Constraints

Game is console based and requires a database. Must be programmed in Java; must use SQL Lite database.

3.5 Software System Attributes

Once again, the game system is rather simplistic and doesn't require over analysis on system. However, if designed correctly it should be modular enough that scaling can be accomplished with minimal work time, if desired.

It is possible that the system might deal with a corrupted dB but handling of this situation has not been defined as of this time.

3.5.1 Reliability

The game is the game and should perform the same each time. System should not produce any behaviors that distort story, change items, rooms, or creatures from what has been identified in specs.

3.5.2 Availability

Game should be available while user system is powered.

3.5.3 Security

NA

3.5.4 Maintainability

System must be modular in design with little coupling of subsystems. This will allow scalability if enhancements are desired.

3.6 Logical Database Requirements

- *All system game objects will be constructed from data in database.*

3.7 Other Requirements

Game should provide replayable experience, where different actions or additional items found create moderately unique game play for each attempt. Battle system will be randomized and are only as necessary as the player chooses. Desire is to have player return for multiple attempts until possible further chapters are developed.