Bilkent University

Department of Computer Engineering

CS319 – Object Oriented Software Project

Project short-name: An Object Oriented Approach to Zork-Like (Text Based) Games

Analysis Report

Project Group 1

Fatih Karaoğlanoğlu

Ege Tokdemir

Emrehan Furkan

İpek Lale

Analysis Report

Oct 15, 2014

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Contents

[1 Introduction 1](#_Toc254263637)

[2 Current System 1](#_Toc254263638)

[3 Proposed System 1](#_Toc254263639)

[1.1 Overview 1](#_Toc254263640)

[1.2 Functional Requirements 1](#_Toc254263641)

[1.3 Non-functional Requirements 1](#_Toc254263642)

[1.4 Pseudo Requirements 2](#_Toc254263643)

[1.5 System Models 2](#_Toc254263644)

[1.1.1 Scenarios 2](#_Toc254263645)

[1.1.2 Use-Case Model 2](#_Toc254263646)

[1.1.3 Object and Class Model 2](#_Toc254263647)

[1.1.4 Dynamic Models 2](#_Toc254263648)

[1.1.5 User Interface 2](#_Toc254263649)

[4 Glossary 2](#_Toc254263650)

[5 References 2](#_Toc254263651)

Analysis Report

Project short-name: An Object Oriented Approach to Zork-Like (Text Based) Games

# Introduction

Text-based games are games which are played with not technologically visual effects but with the players’ imagination. It uses text charecters instead of graphics. We inspired from the text-based game ‘Zork’(1977) and aimed to remake it by adding it new features and our creativity.

# 2 Current System

Actually there are many text-based games on the market. One of the most known ones is called ‘Zork’. It is an interactive fiction computer game which was written in 1977 using the MDL Programming Language\*\*\*. The game begins in a house. Then player give some directions by writing according keywords on the terminal of the game and game goes on according to these directions. (figure 1)

# Proposed System

What we will done in this project is highly similar to Zork. We will change places and we will add some features. For example in Zork there is a comment ‘read’. If user enters it like ‘raed’ Zork does not understand it and only say; ‘I don’t know this word’. (figure 2) Our game will understand such typos and it will detect the command. In addition Zork is located on 2-dimensional plane. However our game will be on 3-dimensional plane which means there will be orientation options which allow to move up and down.

## Overview

This is how you cross-reference something from the list of references [1]. As the number of a reference changes, use right-click | “Update Field” to update it.

# Functional Requirements

* In the game there must be instructions screen which includes description of the game and keywords to proceed in the game.
* The game is controlled by mouse and keyboard.
* The game must have a score screen which shows the player’s score.
* The game should have tolerance for typos.
* The user should be able to save his last condition in the game.

## Non-functional Requirements

* The response time for commands should be less than 1 second.
* The game should be able to run in Windows XP/Vista/7/8, Mac OS and Unix.
* The content of the game should be easily understandable for the user.
* There should be separate classes for each construction( locations, items, actions etc.) in the game to ease testability and increase flexibility of the system (working with separate classes make easier to add features to the system).

**3.4 Pseudo Requirements**

1. The project should be completed within the 3 months.
2. Java should be usen as a programming language.

System Models

### Scenarios

A hyperlink is here.

### Use-Case Model

A code segment is below:

for (i=1; i<=5; i++)

System.out.println(“report to write”);

If you need to inline code, use “this” style.

### Object and Class Model

Table 1 is an example table.

Table 1 An example table

|  |  |
| --- | --- |
| Key | Value |
| key | Value |

### Dynamic Models

### User Interface

# Glossary

Glossary for any domain-specific terms you use in your report.

# References

1. Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2004, ISBN: 0-13-047110-0.