Magic Realm (1979)

Team 6

Couturier, Stefan

Lam, Ryan

Sayyeau, Christopher

Table Of Contents

1. Introduction
   1. Motivation
   2. Terminology
2. Game Rules
3. Requirments
   1. Functional Requirments
   2. Non-Functional Requirments
   3. Assumptions
4. Use Cases
   1. Use Case Diagram
   2. Use Cases
   3. Responsibilities
   4. Unbounded Use Case Maps
   5. Bound Use Case Maps
5. Design Descisions
   1. Decisions
   2. Structural Model (Magic Realm)
   3. Structural Model (Bots)
6. Object Specifications
7. Interaction Diagrams

1 - Introduction

Magic Realm was first realeased in 1979 by Avalon Hill. It was designed as a fantasy adventure board game by Richard Hamblen. It is a very complex role-playing war game that can be played from 1-16 players over the course of several hours. The rules were rereleased in a second edition in 1986 and a third edition was released by fans after the company went out of business in 1998.

* 1. - Motivation

This report will briefly cover our interpretation and implementation of this board game. It will outline the functional and non-functional requirments of the software, as well as its use cases and the assumptions made throughout its construction.

The goal of the project was to turn Magic Realm into a networked computer game. The official 3rd edition rules were followed as closely as possible to create the software.

* 1. - Terminology

A number of key terms are used throughout this report, which are listed alphbetivally in this section. Definitions are included, however if the term is covered in the Game Rules it will be covered in its own section (2).

|  |  |
| --- | --- |
| Term | Definition |
| Chit  Example 2 | This is a token with symbols on it representing some tangible object in the game universe.  Example 2 |

SEE EXAMPLE

1. - Game Rules

The official third edition game rules for Magic Realm.

IMAGES/PAGES SEE 7Wonders EXAMPLE

1. – Requirements

This Section contains what features must be implemented. Each requirement has its own identifier and a short description. The source of traceability may either be from the official game rules, assumptions, other requirements, or team decisions.

* 1. – Functional Requirements

Functional requirements define what behavior and functionality the software must have. They have been categorized below, based on different game states or areas of functionality.

TABLE SEE EXAMPLE

* 1. – Non-Functional Requirements

Non-functional requirments deal with quality of the program. They are aspects which cover accessibility, usability and maintainability.

TABLE SEE EXAMPLE

* 1. – Assumptions

While building the software many assumptions were made about the rules so that the rules could be captured the software requirments. Many of the requirments above will trace to these items. The following table will identify them along with their justification.

TABLE SEE EXAMPLE

1. – Use Cases

A use case covers a scenario. It details the path of events needed for this scenario.

* 1. – Use Case Diagram

DIAGRAM AND EXPLANATION

* 1. – Use Cases

Each use case is detailed in its own table, which describes its sequence of events.

TABLES SEE EXAMPLEs

* 1. – Responsibilities

These are created from the use cases above; they will be used in the use case maps in section 4.4 and 4.5. The reference for the respective use case is on the right.

TABLE SEE EXAMPLE

* 1. – Unbounded Use Case Maps

These maps correspond to the use cases in section 4.2. Triggering and resulting events are labeled in each following table, as well as the associated responsibilities for ease of viewing.

SEE EXAMPLE

* 1. – Bound Use Case Maps

These maps correspond to the previous unbound ones in the section above. Following each map are the descriptions for triggering and resulting events.

SEE EXAMPLE

1. – Design Decisions

This section will cover the design decisions that were taken with respect to classes and objects chosen for the system. The UML diagram will be in section 5.2.

* 1. – Decisions

SEE EXAMPLE

* 1. – Structural Model (Magic Realm)

SEE EXAMPLE

* 1. – Structural Model (Bots)

This diagram involves the implementation of artificial players in the software. ??!!PRETTY SURE WE DON”T USE THESE

SEE EXAMPLE

1. – Object Specifications

SEE TABLES, IN EXAMPLE

1. – Interaction Diagrams

Below are the UML Unteraction Diagrams. Each correspond’s to the previous bound use case maps in section 4.5.

SEE TABLES, IN EXAMPLE