Fall 2024

COMP 302

gameName

R2M2

Group: groupName

Enes Ak – 80090

Muhammed Babelli - 84342

Yusuf Cemâl Karataş - 83639

İbrahim Cebecioğlu - 79906

Caner Kösem - 80246

Cemal Nişan – 79158

Table of Contents

Use Case Narratives

UCN 1: Place Structure

Use Case Name	Place Structure
Scope	Build Mode
Primary Actor	Player
Description	This use case describes the process by which the player places structures in specific locations on the grid during build mode. Structures serve strategic purposes, such as hiding runes or influencing gameplay. The player is responsible for meeting the minimum structure requirements for each hall.
Pre-Condition/s	• The game must be in build mode. • The selected tile must be empty
Post-Condition/s	The chosen structure is placed on the selected tile in the grid, and the hall layout is updated.
Main Success Scenario (Basic Flow)	1. The player opens the Structure Storage to view available structures. 2. The player selects a structure from the storage. 3. The player chooses a specific tile in the hall grid to place the structure. 4. The game verifies that the selected tile is valid (i.e., not occupied by another object or outside the grid boundaries). 5. If the tile is valid, the structure is placed on the grid, and the hall's layout is updated to reflect the change. 6. A visual confirmation is displayed, showing the structure in its new position.
Extensions (Alternate Flow/s)	4a) **Tile Occupied:** • The game checks the selected tile and finds it already occupied. • The game prevents the structure from being placed.

UCN 2: Rune Collected

Use Case Name	Rune Collected
Scope	Search Rune Controller
Primary Actor	Player
Description	This use case describes the process by which the player interacts with a structure in the hall to determine if it contains a rune. The player must navigate the hero to a position adjacent to the structure and interact with it to uncover its contents.
Pre-Condition/s	• The game must be in play mode. • The hero must be adjacent to the structure being explored.
Post-Condition/s	If the structure contains a rune, it is revealed, and the door to the next hall is unlocked.
Main Success Scenario (Basic Flow)	1. The player navigates the hero to a tile adjacent to a structure on the grid. 2. The player clicks on the structure to initiate exploration. 3. The game verifies that the hero is in a valid position adjacent to the structure. 4. The game checks if the structure contains the rune. 5. If the rune is found, it is revealed, and the door to the next hall is unlocked. 6. A sound effect and visual animation indicate that the rune has been discovered.
Extensions (Alternate Flow/s)	3a) **Hero Not Adjacent to Structure:** • The game detects that the hero is not in a valid position to interact with the structure. • The game prevents the interaction. 4a) **Structure Does Not Contain Rune:**

	• The game checks the structure and finds no rune. • The player must continue exploring other structures to locate the rune.

Domain Model

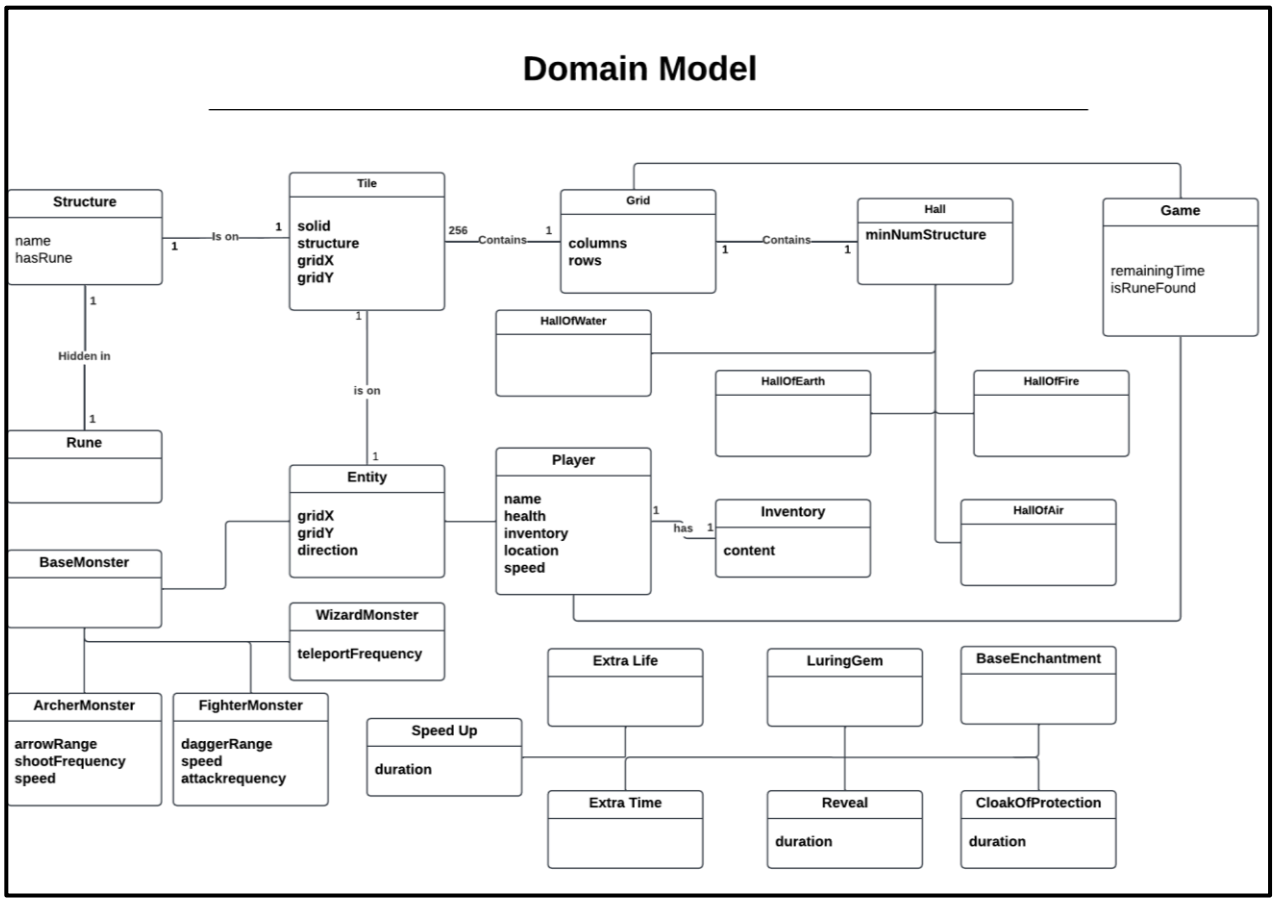


Figure 1: Domain Model

System Sequence Diagrams

SSD 1: saveGame

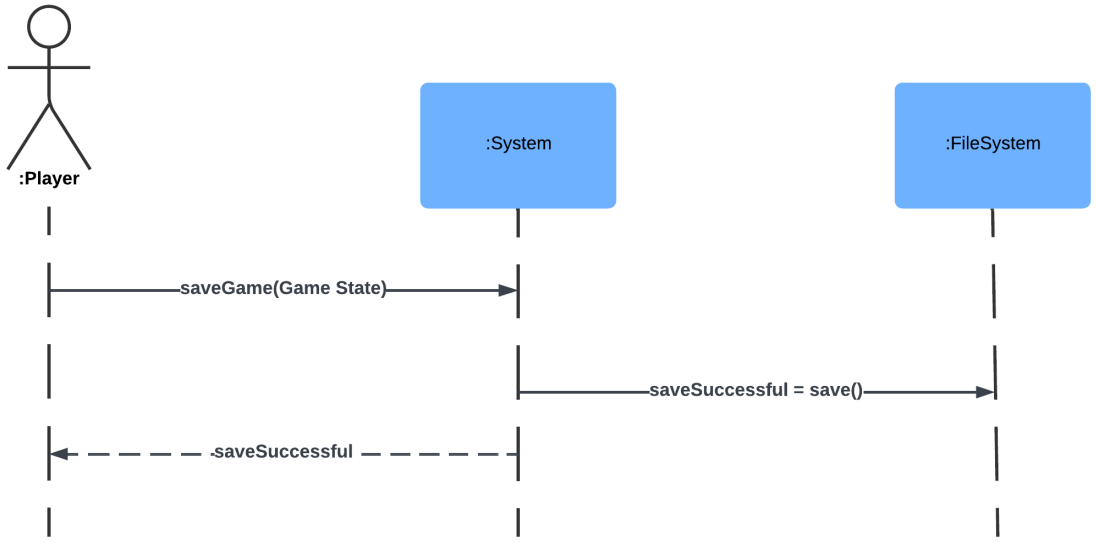


Figure 2: SSD 1 saveGame

SSD 2: loadGame

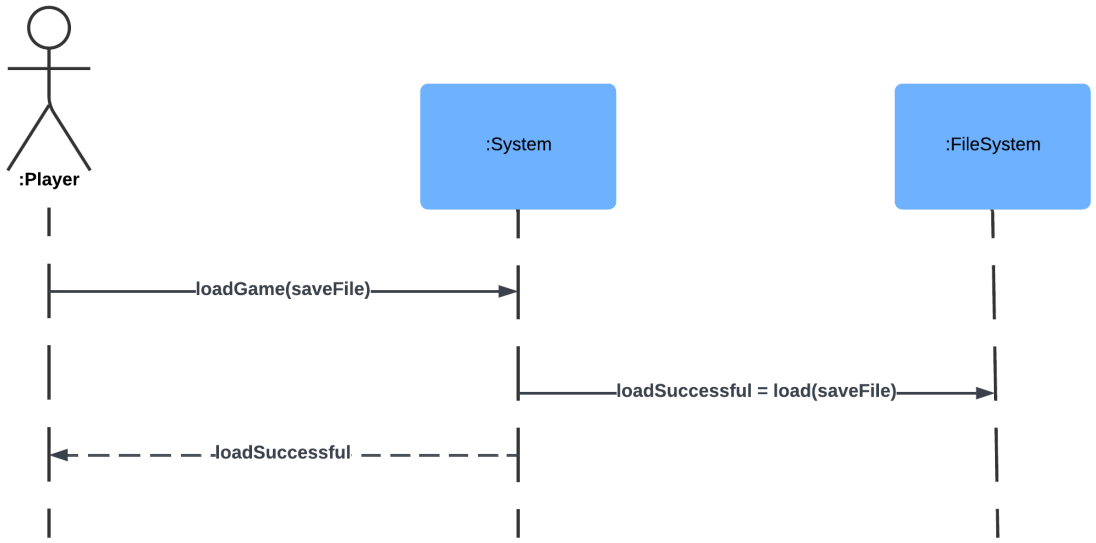


Figure 3: SSD 2 loadGame

Operation Contracts

OC 1: runeCollected

Operation	runeCollected(Tile: ClickedTile)
Cross References	Use Cases: Explore Structure
Preconditions	• The Game is in the Play Mode. • There is a Structure in *location* in question. • Player must be within the one tile of the Structure.
Postconditions	• The rune was revealed if it was in this *structure.*

OC 2: exitGame

Operation	exitGame()
Cross References	Use Cases: Exit the Game
Preconditions	• You are already in the game
Postconditions	• The game was terminated

Supplementary Specifications

Version	Date	Description	Author
Inception Draft	November 20, 2024.	First draft. To be refined during further development phases.	Project Team

Introduction

This document serves as the repository of all requirements for the "*gameName"* game that are not explicitly detailed in the use cases. It outlines essential functionality, usability standards, and non-functional requirements to ensure the development of a robust and engaging game. While the use cases focus on gameplay mechanics, this specification expands on the foundational elements required for a seamless user experience and scalability of the game.

Functionality

Logging and Error Handling

All errors, including invalid grid movements or overlaps, should be logged for debugging purposes. Persistent storage should record these errors for further analysis and future improvements to the game.

Pluggable Rules

The game should support the ability to customize gameplay rules, such as modifying the time limits or adjusting difficulty settings, to enhance replayability and adapt to player preferences. These rules may be defined and executed at various points within gameplay.

The game supports saving and loading, so the players can save their progress while playing the game and load a save file to continue later.

Security

The game ensures that players cannot access the source code to manipulate it.

Usability

Human Factors

The user interface should be clear and easy to navigate for players of all skill levels. Game elements such as the player's inventory, remaining time, current hall, and player's lives must be prominently displayed and easily visible during gameplay.

The use of colors should be mindful of accessibility needs, such as providing a colorblind mode. The game should use alternative visual cues or textual hints for important events (e.g., finding a rune or losing a life) to ensure clarity for all players.

The gameplay experience should be smooth and intuitive, emphasizing strategic thinking and exploration. Audio and visual feedback, such as sound effects when the rune is found or time is running out, should enhance the immersive experience. Pausing, resuming, or exiting the game should be simple and immediately effective.

Performance

The game must maintain a smooth user experience throughout, ensuring that all animations and interactions are seamless. Gameplay should run without delays, even when multiple monsters, objects, or enchantments are active in the hall. Loading times between halls or transitions should be minimal to maintain player immersion.

Glossary

Player: The main character controlled by the player. The player explores the dungeon halls, searching for runes while avoiding monsters and collecting enchantments. The player begins the game with three lives, which can be replenished by collecting specific enchantments. The player is the key figure responsible for unlocking each hall and eventually escaping the dungeon in a limited time.

Rune: A magical artifact hidden within each hall. The player must locate the rune to unlock the exit and proceed to the next hall. Runes may be teleported to new locations by Wizards, adding an extra layer of challenge to the game.

Halls: The dungeon consists of four themed halls—Earth, Air, Water, and Fire. Each hall must be completed in sequence by finding its hidden rune within the given time limit. The design and object placement in each hall differ, designed by the player and also, each hall has their own requirements.

Structure: A type of object in the game that is placed on tiles within the grid. Structures serve as obstacles or strategic elements in the game. They can hide runes or influence the player's navigation, requiring players to plan their movements and actions around their placement.

Tile: The basic unit of the grid-based map in the game. Each Tile can host a structure, the player, a monster, or an enchantment. The player can move only on empty tiles. Tiles serve as the foundation for strategic movement and structure placement in the game.

Monsters: These are enemies that obstruct the player's progress and create challenges. There are three types of monsters:

1. **Archer Monster**: Shoots arrows every second if the player comes within four squares. The player can avoid detection by using the Cloak of Protection enchantment.
2. **Fighter Monster**: Attacks the player with a dagger when adjacent. It can be distracted using the Luring Gem enchantment.
3. **Wizard Monster**: Does not attack the player but teleports the rune to a random location every 5 seconds, making it harder to locate. Depending on the remaining time, the wizard monster will act differently, making the game easier as the remaining time runs out.

Enchantments: Special items that assist the player in various ways. They appear randomly in the halls and provide advantages like extra time, extra lives, or strategic abilities. Some enchantments are used immediately upon collection, while others can be stored in the player's inventory for later use.

Inventory: The player's inventory for storing collected enchantments. Stored enchantments can be used strategically during the game to overcome specific obstacles or challenges.

Grid: The dungeon map is divided into a grid of tile. Each tile can contain one object, such as the player, a monster, a wall, or an enchantment. The grid structure ensures organized and systematic movement for all game elements.

Timer: A countdown mechanism that sets a time limit for each hall. If the timer runs out before the rune is found, the game ends. Collecting the Extra Time enchantment can add seconds to the remaining time.

Build Mode: The phase where players design the interiors of the dungeon halls before starting the game. Players must place a minimum number of objects in each hall, creating opportunities for rune hiding and strategic gameplay.

Play Mode: The active phase of the game where the player explores the halls, avoids monsters, collects enchantments, and searches for the rune. The player's progress, remaining lives, and timer are displayed during this mode.

Luring Gem: A specific enchantment used to distract Fighter monsters. The player throws the gem in a particular direction, causing nearby Fighter monsters to follow it, allowing the player to move away safely.

Cloak of Protection: An enchantment that hides the player from the Archer monster's vision, enabling close proximity without losing lives. The effect lasts for a limited duration.

Extra Life: An enchantment that immediately adds one life to the player upon collection. This helps the player continue their journey if lives are lost during encounters with monsters.

Extra Time: An enchantment that adds additional seconds to the timer when collected. This effect is applied immediately upon collection, giving the player more time to search for the rune and complete the current hall.

Reveal Enchantment: A helpful enchantment that highlights a 4x4 area on the grid where the rune is hidden. The player can use it to narrow down the search area strategically.

Pause/Resume Button: A feature in the game interface that allows the player to pause the game at any time during play mode. The game can be resumed from the same state.

Main Menu: The initial screen that appears when the game is launched. It provides options to start a new game, access the help screen, or exit the game.

Help Page: A section of the game's interface that provides players with detailed information about gameplay mechanics, controls, objectives, and descriptions of game elements such as monsters, enchantments, and runes. It is designed to guide new players and assist them in understanding how to play the game effectively.

Extra Features

Random Map Initilization

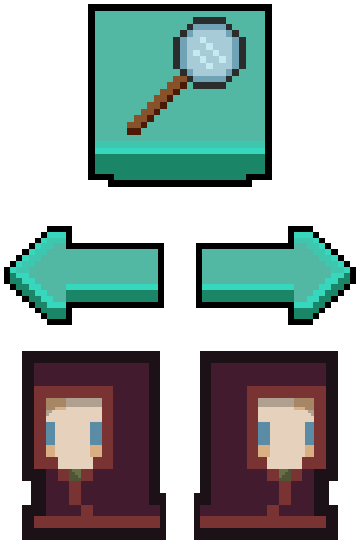
In the build mode, a dice icon has been incorporated, allowing players to initialize the hall with randomly placed structures of various types. This feature streamlines the process by enabling players to save time and effort, such as avoiding the need to place 17 structures individually. To use the feature, the player must click on the dice icon and then select a specific grid, after which the required number of structures will be randomly placed within the designated area. Since the player remains in build mode after using the dice, they retain the flexibility to edit or reposition the randomly placed structures as desired.

New Enchantment: Speed Up!

This enchantment can be stored in the inventory and, when activated, doubles the player's speed. This adds greater excitement and strategic depth to the game.

Custom Pixel Arts

We have designed a collection of pixel art specifically for our game:



Sound Effects

We incorporated music and various sound effects into our game, some of which we created ourselves. For instance, "archer.wav" plays when the archer shoots an arrow, while "structureSounds" is triggered when the player clicks on a structure without a rune.

Try Again Button

If the player wins the game, a victory screen is displayed, whereas a game over screen appears if the player loses for any reason. Both screens include a "Try Again" button, which allows the player to restart the game quickly with the same hall setups upon being clicked.