Software Requirements Specifications

COS 301: REDIRECTION

2018

Team Members

Stephen Teichert - u16254661

Kyle Wood - u16087993

Russell Dutton - u16016612

Jeffrey Russell - u16010648

Justin Grenfell - u16028440

Byron Antak - u16039689

Contents

1 Project Overview

1.1 Purpose

The system should allow a user to be able to create custom bot scripts for the game DOTA 2, to ensure that the bots follow a particular strategy so that the user may practise against that strategy and learn how to play against such a strategy.

1.2 Project Scope

The aim of the project is to design system that gives users the ability to configure bot behaviour on a micro and macro level (using an intuitive and user-friendly interface), where this behaviour is described into LUA files according to the DOTA 2 Scripting API to ensure that bots perform as desired.

The system is still under development and no name has been decided upon for the product. The system will provide a simple interface to customize bot behaviour according to a predefined set of behaviours and actions. The system, however, will **NOT** allow the user to specify exact behaviour (such as always moving in zigzags). Rather, it will allow the user to specify generic behaviour. We are not sure exactly how far we will get with this system and as a result we have decided that we will have the following as non-negotiable.

For team level options, the following will be possibilities:

- Whether the bots want to push early for a win, or delay until late-game, which will be controlled by when the bots want to farm, push and delay lanes. E.g. Bots wanting to delay would likely want to defend lanes more than push them out, and farm a lot
- When to attempt to kill Roshan to acquire the Aegis
- When to buy back to defend a lane
- If and when to roam to gank other heroes.

Should we complete the above well before the end of the project, we will attempt to include the following:

For each hero the user will be able to specify:

- A custom Ability progression
- A custom Item progression
- A custom Talent progression
- Lane Assignment (Top, Bottom or Middle)

1.3 Definitions, acronyms and abbreviations

Terminology and Concepts Dota/Dota 2 - Defense of the Ancients 2, the game for which we are designing the bot configuration system.

Ancient - The core building of each team which must be protected at all costs. Losing this building loses the game.

Bot - A computer controlled hero in a team in Dota.

Radiant - The team on the bottom left hand side of the Dota map.

Dire - The team on the top right hand side of the Dota map.

HP - Hit Points, the amount of sustained damage a hero can take.

MP - Mana Points, used for casting spells.

Lane - One of three paths on the Dota map. The lanes are located on the Left/Top side of the map, Middle and Bottom/Right side of the map and are referred to as top, mid and bottom or bot lanes, respectively. Lanes can also be classified into 3 types: Safe Lane, Mid Lane and Off lane. The Safe Lane has its Tower placed furthest from the base to provide additional support for the heroes in that lane. For Radiant, Safe Lane is the bot lane. For Dire, it is top lane. Radiant and Dire have their mid towers placed mostly equally apart on each side of the map, providing a very equal match-up between the heroes in the lane. Offlane is the hardest lane, because the tower is closest to the base and the activity in the lane is likely futher away from this tower in this lane than the other team. For Radiant, Offlane is top lane and for Dire it is bottom lane. Note that a safe lane for one team faces the offlane of the opposing team.

River - The river that divides the map into the Radiant and Dire sides.

Jungle - The areas on the map that are not a lane and not in the river.

Objective - An entity in the Dota game that is one of many steps in achieving victory. An objective can be a Tower, a Barracks or Roshan.

Roshan - A massive ancient creep that grants bonuses to the heroes that kill him, including Aegis of Immortals, Cheese and a Refresher Shard. The Aegis will grant any hero who wields it a second life; the cheese will instantly restore 2000 HP and MP, and the Refresher Shard is a one-time use item that instantly refreshes all cooldowns of a hero.

Tower - A defensive turret that can be destroyed by the opposing team. There are 3 turrets per lane, each placed more or less the same distance apart in the

lane, but distances do vary. These towers are referred to as Tier x towers, where x is either 1, 2 or 3 with a Tier 1 tower being furthest from base and Tier 3 being closest to base. There are an additional 2 towers in front of the team's Ancient, called Tier 4 towers.

Teamfight - A big fight involving nearly all team members of both teams in any arbitrary location on the map. The victor of the teamfight is determined by the number of losses, and if it is even, it is then decided by the number of core heroes still alive, and if it is still even, then it is either decided by the net gold change of a team, or it is dubbed a draw. Victory in a teamfight will likely result in an objective being taken, whereas defeat will result in a very defensive stance being taken by the team.

Creeps - Little minions that spawn and run down a lane to meet the opposing team's creeps, also called a creep wave. There are also neutral creeps that spawn in the jungle areas of the map that give gold and experience when killed. If a team destroys a barracks of the enemy team, then the creeps of the team that destroyed the barracks will start spawning stronger, upgraded creeps in that lane, called super creeps. Once all barracks are destroyed, the creeps are further upgraded to mega creeps. An upgraded creep has more HP, and gives less gold and experience.

Push - a collective effort from a team to attempt to advance down a lane to attempt to take an objective, a teamfight, or both.

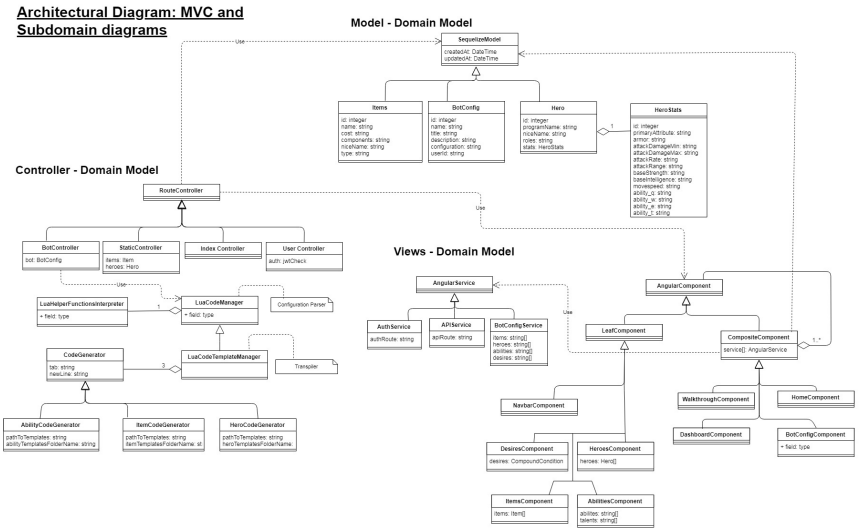
Farm - The simple activity of a hero collecting gold and experience by killing creeps.

Defend - The act of staying close to or even behind your own tower and creep wave when the enemy team is pushing down your lane.

Roaming - the act of not remaining in a lane for too long, but rather moving between lanes to attempt to kill enemy heroes with the other heroes in that lane or in the jungle.

Ganking - The act of a group of team players grouping together to attempt to kill enemy heroes on the map.

1.4 UML Domain model



Models.jpg

2 Functional Requirements

2.1 Users

We expect have two types of users:

2.1.1 Novice DOTA 2 Player (referred to as novice)

This user is inexperienced in DOTA 2 and knows very little about the game as whole. This user will need to have as much information about the game and system provided to them so that they can understand how to use the system. This user would prefer a user friendly interface so that they are not overwhelmed with the information and so that using the system is easy to understand despite their lack of knowledge. This user will also need to know how to install the bot scripts once generated for them as well as.

2.1.2 Intermediate - Advanced DOTA 2 Player

This user is experienced with DOTA 2 and knows the concepts of the game quite well and hence will not need as much information provided to them. This user would like to use the system but would prefer to have it be as quick as possible while still being intuitive.

2.2 Subsystems

2.2.1 Database Manager

The database manager subsystem controls everything related to the database, anything that gets put in the database will go though this subsystem

2.2.2 Transpiler (Back-end)

The transpiler is the system that converts the object into LUA code to make it compatible with the game DOTA 2. It receives an object of all the user inputs from the frontend.

2.2.3 Intermediate Format Analyser

The intermediate format analyser is the subsystem that receives the raw user input from the front-end and converts it it into an object that the backend systems will understand and then sends it to them for further action

2.2.4 Front-end (Angular)

The front-end is the user interface, this is the system that the user will interact with. It receives all the user input and sends it to the intermediate format analyser for interpretation