UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO E ENGENHARIA DE COMPUTAÇÃO

ALUNO(S) AUTORES(S)
HENRIQUE CORREA PEREIRA DA SILVA
GABRIEL STEFANIAK NIEMIEC
MARIA CLARA MACHRY JACINTHO

RELATÓRIO SCIPIO PROJECT

Relatório apresentado como requisito parcial para a obtenção de conceito na Disciplina de Modelos de Linguagens de Programação

Prof. Dr. Lucas Mello Schnorr
Orientador

Porto Alegre
2018

SUMÁRIO

1 INTRODUÇÃO

O problema escolhido pelo nosso grupo consiste em desenvolver um software (*bot*) que minere recursos, desenvolva unidades, e conquiste os adversários no jogo Starcraft Broodwar. Devemos fazê-lo de duas maneiras diferentes: usando o paradigma funcional e usando o paradigma de orientação a objetos.

A linguagem escolhida para a implementação do projeto nos dois paradigmas foi *Scala*. O principal motivo para essa escolha foi a compatibilidade com a *Brood War API* (BWAPI), uma interface de programação que decidimos utilizar na solução do problema. Mais detalhes sobre a linguagem se encontram no Capitulo 3.

2 VISÃO GERAL DO PROBLEMA

Jogos de estratégia em tempo real (*RTS*) derivam de jogos de tabuleiro como xadrez ou damas. As principais características de um RTS são:

- **Simultâneo**: Jogadores tomam decisões ao mesmo tempo
- **Tempo-real**: Não existem turnos, os jogadores podem realizam movimentos independente do tempo que se passou ou do adversário.
- **Parcialmente observável**: Os jogadores não tem a visão completa do tabuleiro
- **Não-determinístico**: Algumas ações envolvem fatores aleatórios
- **Grande quantidade de estados possíveis**: Cada unidade no jogo pode realizar diversas ações em diferentes locais do mapa, ocasionando uma enorme possibilidade de estados possíveis durante um jogo.

O jogo possui três raças diferentes:

- **Protoss**: Possuem unidades mais caras porém mais fortes.
- **Terrans**: Possuem unidades com preço médio e força média
- **Zerg**: Possuem unidades de baixo custo porém mais fracas

3 VISÃO GERAL DA LINGUAGEM

Scala é uma linguagem multi-paradigma que foi projetada para expressar padrões comuns de programação de uma maneira elegante, concisa e *type-safe*. A linguagem, além disso, une recursos de linguagens orientadas a objetos e funcionais ao mesmo tempo.

3.1 Características

- **Orientada a Objetos**: Na linguagem, todo o valor é um objeto, cujo comportamento é definido por classes e interfaces.

- **Funcional**: A linguagem também é funcional no sentido que toda função também é um valor. Além disso, define uma sintaxe simples para *funções de alta ordem*, *funções aninhadas*, *currying* e *pattern matching*.

- **Estaticamente tipada**: A linguagem é equipada com um sistema de tipos expressivos que exige que as abstrações oferecidas sejam utilizadas de uma maneira segura e coerente. Não obstante, também possui um sistema de inferência de tipos, para evitar a utilização de definições redundantes de tipos.

- **Utiliza a JVM**: Projetada com a ideia de interoperar com o *JRE*, a linguagem usufrui de uma larga opção de bibliotecas Java por compartilhar do mesmo modelo de compilação de Java e da performance já estabelecida (e, ao longo dos anos, construída) da máquina virtual.

3.2 Benefícios

A linguagem escolhida detém várias facilidades nos mais diferentes aspectos. Primeiramente, por compilar em Java Bytecode, podemos utilizar das mais variadas bibliotecas da já consagrada, amada e adorada (e universalmente odiada) linguagem Java. Não somente pode utilizar abstrações diretamente de Java, também trás consigo uma sintaxe simples e concisa para ambas **orientação a objetos** e a **programação funcional**.

3.3 Principais Aplicações

Dentre os usos populares de Scala, podemos citar:

- Programação concorrente e distribuída
- Análise de dados
- Aplicações web
- Libraries
- Scripting simples

4 RECURSOS

5 ANÁLISE CRÍTICA

6 CONCLUSÃO

REFERÊNCIAS