

# PokemOz

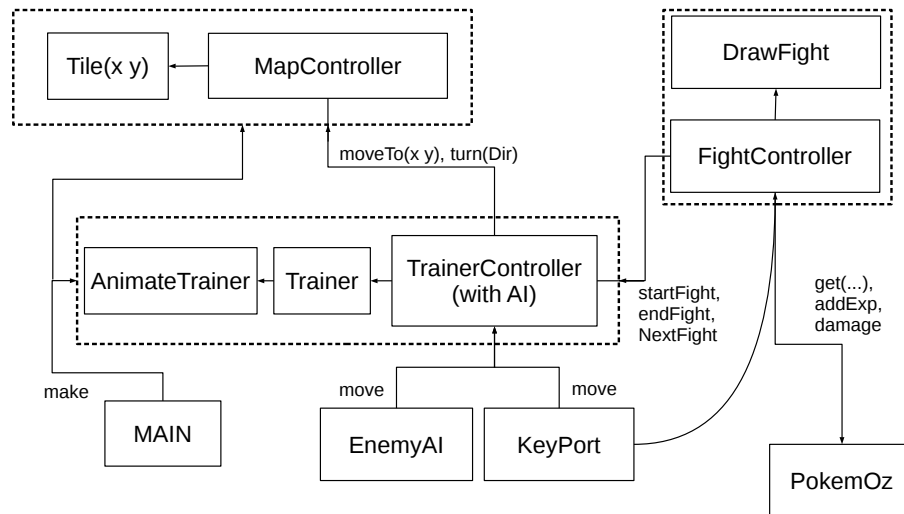
Victor Joos and Antoine Vanderschueren

## Introduction

We were tasked with the import mission of creating a PokemOz game using message-passing concurrency. In this report, we will try to explain the design decisions and the concepts used along this project.

## Component Diagram

Our components and thus our component diagram are based in a large part on the lift example in section 5.4 of CTMCP <sup>1</sup>.



Every one of these components are modeled using **NewPortObject** or an alternative **NewPortObjectKillable** which allows the game to stop the thread when it is no longer needed, to save on resources.

## State Diagrams

We programmed every part of our program using port-objects. Using these port-object allowed us to make an easily testable system using states. In the

<sup>1</sup>VAN ROY, P., HARIDI, S., *Concepts, Techniques, and Models of Computer Programming*, The MIT Press, Cambridge.

following section, we will show the state diagrams for the stateful port-objects.

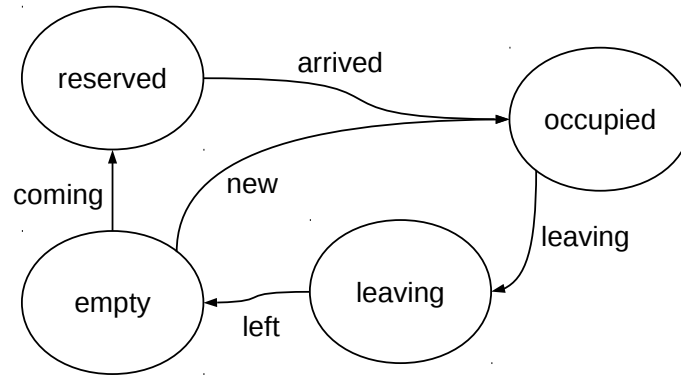


Figure 1: Tile State Diagram

**Tile** A Tile on the map has an easy state diagram. Each tile has a set of fixed coordinates that can be used by other port-objects to send a tile some messages, through the MapController. The **reserved** and **leaving** intermediate states allow a tile to refuse new Trainers wanting to go on a tile while another trainer is not yet on the tile, but is animating to it at the moment.

**PlayerController** This state diagram shows the states of both the PlayerController and the Trainer port-objects. The TrainerController receives “keys” from the keyboard or the Artificial Intelligence, and will then move the Trainer on the map or start a fight, and relinquish control to the FightController.

**FightController** The last important port-object is the FightController. Every time a fight takes place a new FightController port-object is created. It waits for input from the user, or uses autofight to let the player fight.

**Artificial intelligence** We also want to attract the readers attention on the artificial intelligence that can be triggered using the `--ai` switch on the command-line. The AI is implemented using Dijkstra’s shortest path algorithm. It calculates a distance between its position and the last tile with a greater distance for the grass and the trainer surroundings.

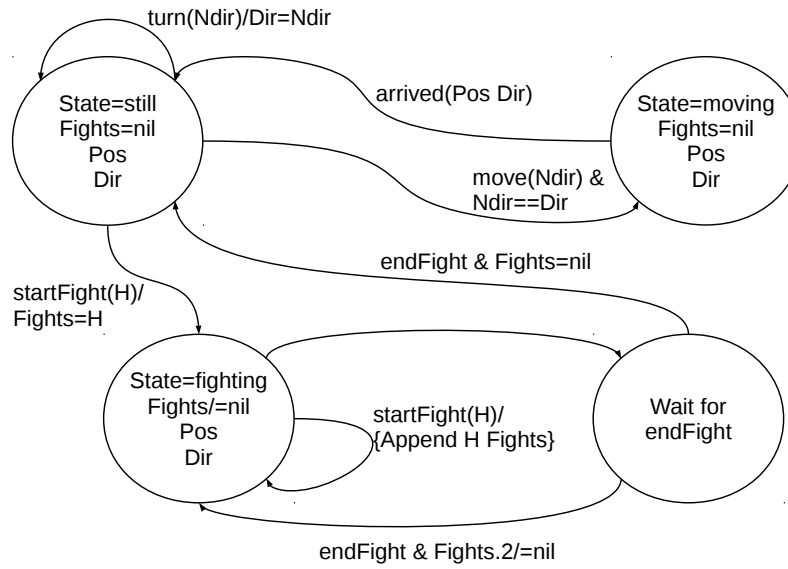


Figure 2: PlayerController State Diagram

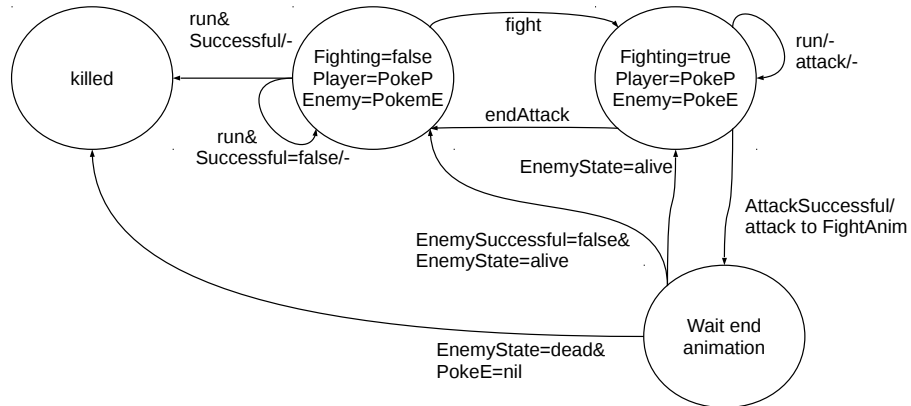


Figure 3: FightController State Diagram