COSC 3P91 – Assignment 3 – 7376726

PARKER TENBROECK, Brock University, Canada

Traffic simulation design and implementation

1 MVC ARCHITECTURAL DESIGN PATTERN

The Model is represented by the `RoadMap` and `SImulation`. These will update the View component (defined later) when either the simulation tick happens or when a frame is drawn.

The Controller components is both a collection of `SimSystems` that define the behavior of all parts of the simulation and also (importantly) `Controllers` attached to `Vehicless` that allow for player input. These controllers will update and change the behavior of the Model(Simulation) depending on what the user has inputted.

The View is represented by either `View` or `TextView` and is updated by the Model every frame/tick.

The output from either(or both) View(s) is interpreted by the User and the User is then allowed to interact with the Controller completing the cycle.

There was no change from Assignment 2 for any of these components aside from renamming `TextDisplay` to `TextView`.

2 FACTORY OR ABSTRACT FACTORY CREATIONAL DESIGN PATTERN

There are two places where Factory design patterns are used, the first is the `VehicleFactory` which gives an interface for constructing arbitrary `Vehicles` for the `SourceIntersection`.

The change made from Assignment 2 was to make my own Interface `VehicleFactory` and not use the default java `Supplier` Interface.

The second place Factory patterns are used is in `XmlTypeInstantiationFactory` These factories make instances of classes whos constructor and class path is defined in the maps XML file to allow for custom Vehicles to be defined and spawned on any `RoadMap` just from the XML file.

These components are new alongside the XML and were not present in Assignment 2.

3 COMPOSITE STRUCTURAL DESIGN PATTERN

This pattern is implemented with the `XmlTypeInstantiationFactory`, This is used to instantiate any java class with any constructor with only XML. Primitives `byte`,`short`,`int`, `long`,`float`,`double`,`char`, `String`, and Objects with empty constructors are treated as Leafs. Objects with non empty constructors are treated as Composites and hold any number of additional composite or primitive Components.

Author's address: Parker TenBroeck, Brock University, 1812 Sir Isaac Brock Way, St. Catharines, ON, L2S 3A1, Canada.

Update date: 22 March 2024

These components are new alongside the XML and were not present in Assignment 2.

4 LOADING AND PARSING XML FILES

XML loading is handled through the `xml` package by `XmlMapTools` and allow for loading and saving XML files to Input/Output streams respectively.

5 DEFINING THE XML SCHEMA IN A XSD FILE

A schema is included and verifies the loaded XML to ensure it is in the format needed by the map loader. It is present in the res folder and called `roadmap`$_s$*chema.xsd*, *Anexceptionisthrownandthes*