МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ М. В. ЛОМОНОСОВА

Факультет Вычислительной Математики и Кибернетики

Отчет о выполнении задания практикума

«Система контроля движения электропоездов»

выполнил:
Филин Максим
424 группа

Москва
2017

Оглавление

1. Уточненная постановка задачи
2. Диаграмма классов
3. Текстовые спецификации классов
4. Инструментальные средства
5. Файловая структура программы
6. Пользовательский интерфейс

1. Уточненная постановка задачи

Рассматривается *линейный участок железной дороги*, соединяющий *N* станций *(7≤ N ≤ 20)*. Известно *суточное расписание* движения электропоездов между этими станциями (в одном направлении), которое включает *M маршрутов (5≤ M ≤20)*. Каждый маршрут фиксирует:

• станцию-пункт отправления и станцию-пункт назначения;
• промежуточные станции маршрута, в которых электропоезд делает остановку;
• время прибытия и отправки электропоезда в каждой станции маршрута.

Фактическое движение электропоездов зависит не только от расписания, но и от некоторых непредвиденных событий, к числу которых относятся задержки поездов на станциях маршрута, а также *аварии поездов* и *повреждения железнодорожных путей* (которые на некоторое время нарушают движение).

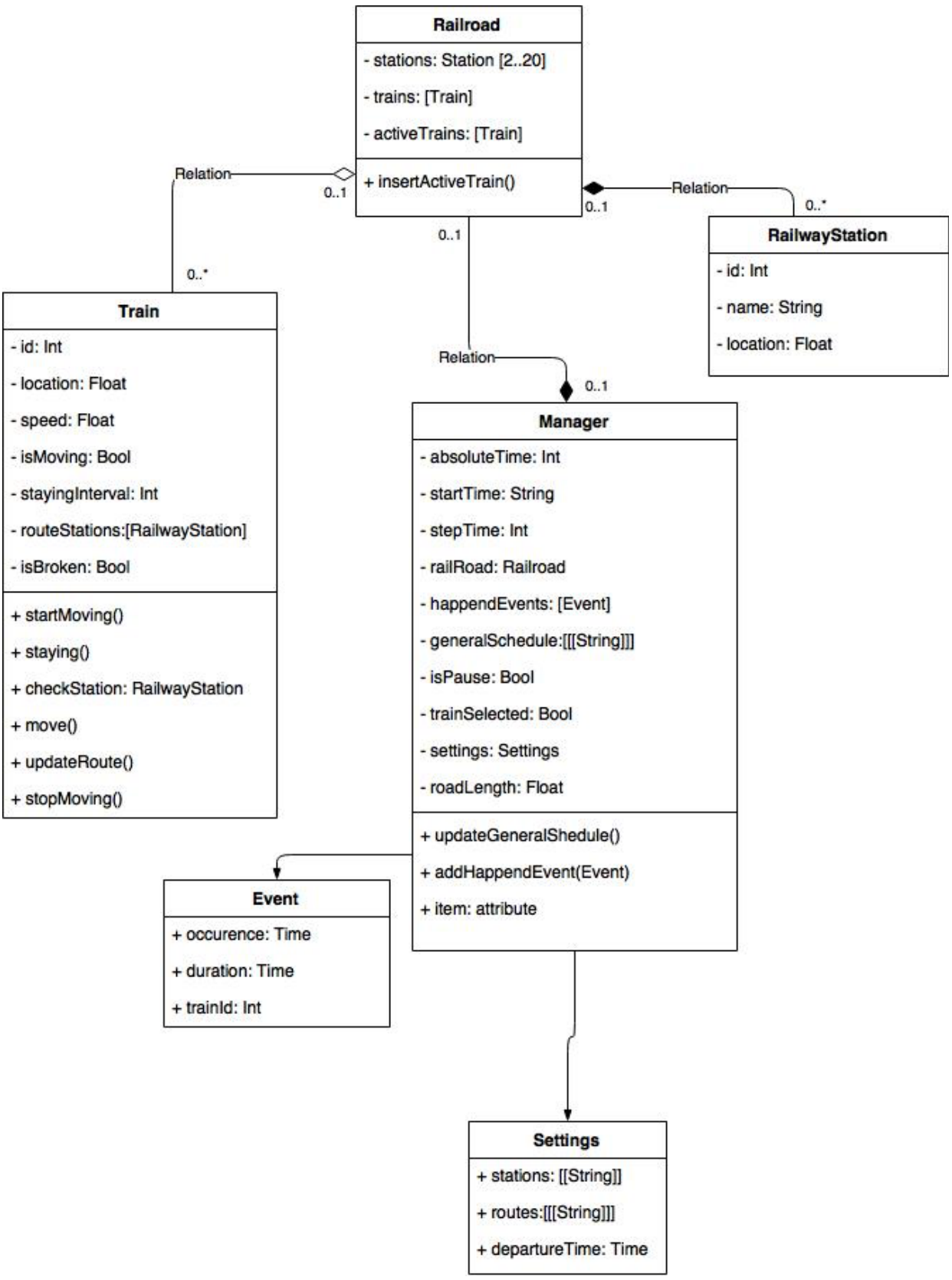
Требуется разработать систему контроля движения электропоездов, которая отслеживает их движение по маршрутам, регистрирует возникающие поломки поездов, а также корректирует при необходимости расписание, определяя время предполагаемого прибытия поездов на каждую станцию маршрута. Можно считать, что электропоезда двигаются по маршрутам с определенной скоростью (например, 70 км/час).

Цель моделирования – изучение стабильности движения поездов при заданном расписании движения в условиях возникающих непредвиденных событий. Период моделирования – один день.

Случайные факторы движения (задержки и аварии) следует моделировать статистически. Случайной величиной является длительность ремонта после аварии.

Кроме величин *N*, *M* и расписания движения, в изменяемые параметры моделирования целесообразно включить: время суток начала моделирования движения электропоездов, шаг моделирования – 15 или 30 минут. В ходе моделирования на экране компьютера должна быть изображена схема рассматриваемого участка железной дороги, на которой показано движение поездов в соответствии с расписанием, возникающие аварии и неисправности путей. По запросу необходимо также показать расписание движения, скорректированное в соответствии с уже случившимися событиями.

2. Диаграмма классов



3. Текстовые спецификации классов

class **Manager** {

//MARK: - Public

//получить текущее время в 24м формате
public func getWorldTime() -> String

//получить время в секундах от начала дня
public func getTime() -> Int

public func updateAbsoluteTimeOnStepTime(withFps fps: Int)

public func sortTrainsByStartTime()

public func insertActiveTrain(train: Train)

//получить время в секундах от времени начала моделирования
public func getAbsoluteTime() -> Int

public func getTrains() -> [Train] {

public func getActiveTrains() -> [Train]

public func getStations() -> [RailwayStation]

public func isOnPause() -> Bool

public func isTrainSelected() -> Bool

public func getStepTime() -> Int

public func getRoutes() -> [[[String]]]

public func getCurrentRoute() -> [[String]]

public func getRoutesNumber() -> Int

public func getTrackView() -> UIView

public func getHappendEvents() -> [Event]

//MARK: - Setters

```
    public func setCurrentRouteNumber(number: Int)

    public func setOnPause()

    public func setOnPlay()

    public func setOnTrainSelected()

    public func setOffTrainSelected()

    public func setStepTime(time: Int)

    public func setTrains(trains: [Train])

    public func setActiveTrains(activeTrains: [Train])

    public func setTrackView(trackView: UIView)

    public func getStationId(stationName: String) -> Int

//MARK: - Private functions

    private func makeRailwayStations(stations: [(String, Float)]) -> [RailwayStation]

    private func makeTrains(routes: [[[String]]]) -> [Train]

//MARK: - Other

    public func updateGeneralShedule(with route: [[String]], trainId: Int)

    public func addHappendEvent(newEvent: Event)

    //MARK: - Static functions

    public static func sum24Time(time1: String, time2: String) -> String

    public static func sub24Time(time1: String, time2: String) -> String

    //функция перевода 24 формата (строка HH:MM) в абсолютное время
    public static func convertToSecTime(time: String) -> Int

    //функция конвертирования из секунд в 24й формат
    public static func convertTo24Time(time: Int) -> String
```

```
    public static func getRoadLength() -> CGFloat

}

class Train {

 //MARK: - Getters

    public func getStationTime(stationId: Int) -> Int

    public func getRoute() -> [[String]]

    public func getTrainView() -> UIView?

    public func getSpeed() -> Float

    public func getId() -> Int

    public func isTrainMoving() -> Bool

    public func getLocation() -> Float

    public func getEndLocation() -> Float

    public func getStartAbsoluteTime() -> Int {
        return self.startAbsoluteTime
    }
    //MARK: - Setters

    public func setTrainView(trainView: UIView)

    //MARK: - Other

    public func printTrain()

    public func removeTrainView()

    public func startMoving()

    public func stopMoving(stayingInterval: Int)

    public func checkStation(stepTime: Int, fps: Int) -> RailwayStation?


    //изменить времена в маршруте на delta (сек)
```

```
    public func updateRoute(on delta: Int)

}

class Event {

   //MARK: - Getters
    public func getTrainId() -> Int

    public func getOccuranceTime() -> Int

    public func getInterval() -> Int
}

class RailwayStation {

    //MARK: - Public

     public func printStation()

    //MARK: - Getters

   public func getId() -> Int

   public func getLocation() -> Float

   public func getName() -> String

   public func getStationTime() -> Int


    //MARK: - Setters

    public func setStationView(stationView: UIView)

    //MARK: - Other

    public func isVisited() -> Bool

    public func makeVisited()

    public func removeStationView()
}
```

4. Инструментальные средства

Язык программирования: Swift 3.0
Среда разработки: Xcode 8.2
Используемая библиотека: UIKit

5. Файловая структура системы

ViewController.swift - файл, в котором происходит управление моделью и обработка событий. В нем определены все визуальные элементы

Settings.swift - файл с начальными данными о маршрутах и станциях на линии

Manager.swift - определение и реализация класса, с помощью которого происходит управление моделированием

Train.swift - определение и реализация класса Train

Event.swift - определение и реализация класса Event

RailwayStation.swift - определение и реализация класса RailwayStation

Railroad.swift - определение и реализация класса Railroad

6. Пользовательский интерфейс

Пользовательский интерфейс представляет собой единое окно для настройки и визуализации модели.

На экране отображены основные параметры и кнопки для их изменения. Основная информация выводится на двух таблицах - таблице Маршрутов и таблице Событий.

Можно регулировать шаг моделирования, масштаб линии железной дороги, среднюю величину интервалов поломок поездов.

Чтобы смоделировать поломку поезда, на него необходимо нажать, после этого он остановится на случайное количество времени и расписание будет обновлено с учетом задержки поезда.

Система контроля движения электропоездов

Время: 15:52

Шаг: 01:10

Старт

Рестарт

Поезд	Время аварии	Задержка

Станция	Время прибытия	Время отправления	Время стоянки
Москва	--	10:30	--
Зеленоград	11:30	11:40	...
Тверь	14:40	14:50	...
Великий Новгород	18:50	19:00	...
Санкт-Петербург	20:00	--	--

Величина интервала поломки: 00:31

Часы шага

Минуты шага

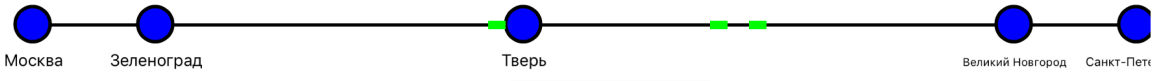
Масштаб: 1.8

Выберите маршрут:

Зеленоград - Великий Новгород

Москва - Санкт-Петербург

Москва - Санкт-Петербург



Система контроля движения электропоездов

Время: 19:14

Шаг: 00:10

Пауза

Рестарт

Поезд	Время аварии	Задержка
Train 1	18:08	00:31
Train 3	18:16	00:31
Train 2	18:23	00:31
Train 1	19:05	00:31

Станция	Время прибытия	Время отправления	Время стоянки
Зеленоград	--	15:02	--
Тверь	18:02	18:12	...
Великий Новгород	22:12	--	...

Величина интервала поломки: 00:31

Часы шага

Минуты шага

Масштаб: 1.8

Выберите маршрут:

Зеленоград - Великий Новгород

Москва - Санкт-Петербург

Москва - Санкт-Петербург