СОДЕРЖАНИЕ

[1. ВВЕДЕНИЕ 4](#_Toc134730998)

[**1.1. Наименование разработки 4**](#_Toc134730999)

[**1.2. Основание для разработки 4**](#_Toc134731000)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 5](#_Toc134731001)

[**2.1. Функциональное назначение 5**](#_Toc134731002)

[**2.2. Эксплуатационное назначение 5**](#_Toc134731003)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 6](#_Toc134731004)

[**3.1. Постановка задачи на разработку программы 6**](#_Toc134731005)

[**3.2. Описание алгоритма и функционирования программы 6**](#_Toc134731006)

[**3.2.1. Чтение информации из файлов 6**](#_Toc134731007)

[**3.2.2. Подготовка к моделированию 7**](#_Toc134731008)

[**3.2.3. Моделирование с помощью клеточных автоматов 7**](#_Toc134731009)

[**3.2.4. Экспорт результатов 8**](#_Toc134731010)

[**3.3. Организация входных данных 8**](#_Toc134731011)

[**3.3.1. Передача файла в программу 8**](#_Toc134731012)

[**3.3.2. Внутренняя организация .shp и .dbf файла 9**](#_Toc134731013)

[**3.4. Организация выходных данных 10**](#_Toc134731014)

[**3.4.1. Вид экспортируемой сводной таблицы 10**](#_Toc134731015)

[**3.4.2. Вид экспортируемого столбчатого графика 10**](#_Toc134731016)

[**3.4.3. Вид экспортируемого графика 10**](#_Toc134731017)

[**3.5. Описание и обоснование выбора и состава технических и программных средств 11**](#_Toc134731018)

[**3.5.1. Описание технических и программных средств 11**](#_Toc134731019)

[**3.5.2. Обоснование выбора технических и программных средств 11**](#_Toc134731020)

[**3.5.2.1. Язык Java 11**](#_Toc134731021)

[**3.5.2.2. Библиотека GeoTools 11**](#_Toc134731022)

[**3.5.2.3. Библиотека JTS 11**](#_Toc134731023)

[**3.5.2.4. Библиотека JFreeChart 11**](#_Toc134731024)

[**3.5.2.5. Технические средства. 11**](#_Toc134731025)

[4. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 12](#_Toc134731026)

[**4.1. Предполагаемая потребность 12**](#_Toc134731027)

[**4.2. Ориентировочная экономическая эффективность 12**](#_Toc134731028)

[**4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами 12**](#_Toc134731029)

[5. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 13](#_Toc134731030)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 14](#_Toc134731031)

1. **ВВЕДЕНИЕ**

## Наименование разработки

**Наименование разработки на русском языке:** Моделирование дорожного движения с высоким разрешением

**Наименование разработки на английском языке:** Very High Resolution Road Traffic Simulation

## Основание для разработки

**Документ, на основании которого ведется разработка:**

Программа выполнена в рамках задания на курсовую работу в соответствии с учебным планом подготовки бакалавров (НИУ ВШЭ, факультет компьютерных наук) по направлению «Программная инженерия». Приказ декана факультета компьютерных наук Национального исследовательского университета «Высшая школа экономики» № 2.3-02/1212-01 от 12.12.2017 "Об утверждении тем, руководителей курсовых работ студентов образовательной программы Программная инженерия факультета компьютерных наук".

1. **НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ**
   1. **Функциональное назначение**

Программа предоставляет высококачественное моделирование дорожного движения на векторном слое дорожного покрытия отдельно взятых проектируемых и существующих районов города. Кроме того, программа предоставляет данные о совершенном моделировании в виде графиков и таблиц с основными характеристиками дорожного движения.

* 1. **Эксплуатационное назначение**

Программа должна использоваться урбанистами и инженерами дорожных схем для выявления проблем с трафиком на будущих и существующих схемах дорожного движения, также разработчиками геоинформационных систем в качестве инструмента для тестирования полученных векторных данных дорожного полотна.

1. **ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ**
   1. **Постановка задачи на разработку программы**

В рамках курсовой работы требуется реализовать исполняемую программу на языке Java для моделирования дорожного движения с помощью клеточных автоматов. Клеточные автоматы будут повышать точность моделирования, обеспечивая высокое разрешение модели. В качестве основы модели должны лежать файлы дорожных карт с расширением «.shp» и «.dbf» подготовленные в программе QGIS. Пример такого файла с картой района Нью-Йорка, о. Манхэттен был получен до разработки программы, на нем предлагается осуществить демонстрацию работоспособности.

Более подробные требования изложены в техническом задании, общие требования предлагались такие:

* Корректно считывать файлы описанных форматов
* Моделирование движения по дорожной сети, основанное на клеточном автомате
* Демонстрация статистических данных
* Экспорт данных в виде графиков и таблиц

Поставленная задача подразумевает разработку программы с консольным интерфейсом. На вход программе будет задаваться путь к файлу, далее с помощью библиотеки GeoTools будет обрабатываться карта и строиться модель клеточного автомата с заданным разрешением с помощью библиотеки JTS. Финальные графики создаются с помощью функционала библиотеки JFreeChart. Также программа будет отдавать на выход растровое изображение самой модели.

* 1. **Описание алгоритма и функционирования программы**

### Чтение информации из файлов

Для работы с геоданными, хранящимися в файлах форматов .shp и .dbf, необходимо создать классы и методы, которые будут использовать библиотеку GeoTools [10]. Прежде чем начать работу с предоставленной информацией нужно ее прочитать и перенести в программу, это выполняют классы описанные ниже.

В рамках этой задачи создано четыре класса:

- **ShpReader** - класс, который будет содержать методы для чтения данных из файлов .shp;

- **ShpRecord** - класс, который будет представлять собой запись(информацию о конкретном объекте дорожного полотна) в файле .shp. Каждая запись содержит геометрию объекта и его атрибутивную информацию.

- **DbfReader** - класс, который будет содержать методы для чтения данных из файлов .dbf;

- **TransportRecord** – Класс для хранения параметров транспортных средств который будет поддерживать свойства, такие как максимальная скорость автомобиля, средняя скорость реакции, источник появления, цель движения. Аналогичен **ShpRecord**, только представляет транспортное средство с атрибутивной информацией.

Методы класса **ShpReader** выполняют функции:

- чтение файла .shp;

- получение информации о геометрии объектов из файла .shp;

- передача этой информации по запросу;

Методы класса **DbfReader** выполняют функции:

- чтение файла .dbf;

- получение информации об атрибутах, связанных с каждым объектом в файле .dbf;

- передача этой информации по запросу;

* + 1. **Подготовка к моделированию**

Для создания клеточной сетки на основе обработанных ранее данных из файла .shp создан класс **Grid** содержащий элементы созданного класса **Cell**, который содержит методы для создания сетки и ее дальнейшей обработки. По сути **Grid** является двумерным массивом заданного разрешения, при этом у каждой клетки есть дополнительные атрибуты, по факту массив трехмерный.

В рамках этой задачи классом **Grid** используются в основном следующие методы:

- readShpRecord - метод, который будет читать данные ShpRecord и конвертировать их в массив значений для генерации растрового изображения;

- createGrid - метод, который создаст и наполнит клеточную сетку на основе массива ShpRecord и растрового изображения;

- calculateCellAttributes - метод, который будет наделять атрибутами каждую ячейку сетки (**Cell**) на основе полученных данных из соответствующего ShpRecord объекта;

- exportGrid - метод, который будет передавать полученную сетку моделирующему классу.

* + 1. **Моделирование с помощью клеточных автоматов**

Процесс моделирования будет происходить в классе **Modelling.**

Для класса **Modelling** был создан отдельный класс **Iteration**, представляющий информацию о состоянии модели на каждый момент ее работы. В его атрибуты собирается относительная загруженность всех клеток, количество передвинутых автомобилей, количество доехавших автомобилей, количество автомобилей на дорогах, и другая информация. Список с экземплярами **Iteration** является результатом моделирования и далее будет передаваться в класс-обработчик результатов.

В данном случае, для моделирования создан класс **Modelling**, который имеет следующие поля:

- roadGrid - экземпляр класса Grid, который представляет собой сетку клеток дороги;

- carsToStart - список автомобилей, которые еще не начали движение. Состоит из экземпляров **TransportRecord,** прочитанных ранее. Этот список изначальный, получается от стороннего класса.

- cars - список автомобилей, которые находятся на дороге. Состоит из экземпляров **TransportRecord,** прочитанных ранее.

- carsFinished - список автомобилей, которые окончили движение. Состоит из экземпляров **TransportRecord,** прочитанных ранее.

- iterationList – результат работы модели, состоящий из экземпляров **Iteration**

Класс **Modelling** имеет следующие основные методы:

- moveCars – основной метод моделирования, который перемещает автомобили на дороге, согласно правилам движения. Например, если автомобили на дороге движутся слева направо, метод будет обрабатывать каждый автомобиль по очереди и перемещать его на следующую клетку. Если на следующей клетке уже есть другой автомобиль, то текущий автомобиль должен остановиться; Вызывается с каждой итерацией моделирования.

- addCar - метод, который добавляет новый автомобиль на дорогу, если тот еще не стартовал так как клетка старта была занята;

- removeCar - метод, который удаляет автомобиль с дороги, если он достиг конечного пункта назначения.

- statCount – метод, собирающий информацию с каждой итерацией моделирования. Каждую итерацию метод добавляет в список iterationList новый экземпляр **Iteration,** с информацией о состоянии модели**.**

**-** exportStat – метод для передачи статистики далее

Кроме этих основных методов, класс **Modelling** может содержать и другие методы, однако они служебные или слишком малы, и на суть алгоритма не влияют.

* + 1. **Экспорт результатов**

Класс-экспортер должен иметь методы, которые принимают список экземпляров класса **Iteration** и используют библиотеку JFreeChart для создания различных графиков, отображающих информацию о состоянии модели на каждый момент ее работы.

Каждый метод должен создавать соответствующий график или чарт на основе переданных данных и настроек, а затем сохранять его в файле с расширением .png, который можно передать дальше для дальнейшего использования или отображения. Графики будут сохраняться на устройстве пользователя, в отдельной директории под каждое моделирование, директории располагаются в корневой папке проекта.

Некоторые из возможных графиков, которые будут созданы:

- График относительной загруженности всех клеток на каждый момент моделирования

- График количества передвинутых автомобилей на каждый момент моделирования

- График количества доехавших автомобилей на каждый момент моделирования

- График количества автомобилей на дорогах на каждый момент моделирования

- Сравнительный столбчатый график самых загруженных клеток, их основные атрибуты

- Сравнительный столбчатый график самых свободных клеток, их основные атрибуты

- Сводная таблица с основной информации по проведенному моделированию (кол-во итераций, машин, средняя загруженность клеток по 9 частям карты по частям света)

Приведены основные графики, которые будет создавать программа при стандартном запуске, однако в интерфейсе будут предложены дополнительные опции.

* 1. **Организация входных данных**
     1. **Передача файла в программу**

При запуске программы, интерфейс попросит у пользователя полный путь до исходных файлов .shp и .dbf. С картой и информацией о трафике соответсвенно.

### Внутренняя организация .shp и .dbf файла

В предложенном файле карты и трафика для модели обязательно должны содержаться такие атрибуты для каждого из объектов:

Таблица 1

Для .shp файла:

|  |  |  |  |
| --- | --- | --- | --- |
| **атрибут** | **возможное значение** | **пример** | **комментарий** |
| b5\_sc | Long | 134110 | Служебная информация |
| b7\_sc | Long | 134110002 | Служебная информация |
| bike\_lane | Byte or NULL | 4 | Количество велосипедных дорожек |
| boroughcod | Byte or NULL | 1 | Код района |
| continuous | Byte or NULL | 1 | Тупиковость улицы |
| created\_by | String | DCP | Создатель |
| date\_creat | String | 29.11.07 | Дата создания объекта |
| time\_creat | String | 00:00:00:000 | Время создания объекта |
| from\_level | Byte | 13 | Высота улицы в начале |
| l\_blockfac | Long | 1222602500 | Служебная информация |
| l\_high\_hn | Long or NULL | 669 | Максимальная высота улицы слева |
| l\_low\_hn | Long or NULL | 623 | Минимальная высота улицы слева |
| l\_zip | Long | 10011 | Почтовый код по левую сторону улицы |
| lsubsect | String or NULL | 1B | Секция в которую попадает левая сторона улицы |
| modified\_b | String | Csarkissan | Кем изменялся объект |
| nonped | Char or NULL | V | Есть ли тротуар (V – тротуара нет) |
| physicalid | Long | 756 | Идентификационый номер |
| r\_blockfac | Long | 1222603317 | Служебная информация |
| r\_high\_hn | String or NULL | 670 | Максимальная высота улицы справа |
| r\_low\_hn | String or NULL | 824 | Минимальная высота улицы справа |
| r\_zip | Long | 11211 | Почтовый код по правую сторону улицы |
| rsubsect | String or NULL | 5A | Секция в которую попадает правая сторона улицы |
| segmentlen | Double | 28,00526474 | Длина сегмента улицы в условных единицах |
| shape\_leng | Double | 54,87169735707 | Длина самой улицы |
| stname\_lab | String | W 17 ST | Название улицы |
| streetwidt | Byte | 44 | Ширина улицы в условных единицах |
| to\_level\_c | Byte | 13 | Высота улицы в конце |
| trafdir | String | FT | Направление движения (FT;TF;TW -двустороннее) |

Таблица 2

Схема .dbf файла

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **атрибут** | **id** | **aggressive\_rate** | **max\_speed** | **start\_physicalid** | **finish\_physicalid** | **reaction\_rate** |
| возможное значение | Long | Byte | Byte | Long | Long | Byte |

Расшифровка значений:

**Long -** числовое целочисленное значение от –9223372036854775808 до 9223372036854775807;

**Byte –** числовое целочисленное значение от –128 до 127;

**Double –** число с плавающей точкой от 4.9e-324 до 1.7e+308;

**String –** массив символов от 0 до 2048 символов;

**Char – ASCII-**символ;

**NULL** – пустое поле

* 1. **Организация выходных данных**
     1. **Вид экспортируемой сводной таблицы**

|  |  |  |  |
| --- | --- | --- | --- |
| **Кол-во автомобилей в модели** | 100 | **Средняя длина маршрута в клетках** | 1276 |
| **Кол-во итераций** | 100 | **Максимальное кол-во автомобилей на карте** | 100 |
| **Средняя загруженность по клеткам** | 100% | **Максимальное время ожидания движения в итерациях** | 10 |
| **Среднее кол-во итераций на проходимость маршрута** | 50 | **Нетронутые клетки** | 234213 |
| **Эффективность движения** | 50% | **Максимальное кол-во автомобилей, проехавших по одной клетке** | 34 |
| **Средняя скорость на маршруте** | 40 | **Общее время на моделирование** | 1234 сек. |

Таблица экспортируется в формате csv с «;» в качестве разделителя. Помимо этих данных, в таблицу может добавляться дополнительная информация о моделировании.

* + 1. **Вид экспортируемого столбчатого графика**

Все описанные ранее чарты буду иметь такую структуру, атрибуты и оси, внешний вид может отличаться в зависимости от платформы запуска.

* + 1. **Вид экспортируемого графика**

Предполагается, что графики будут выглядеть вот так, стоит заметить что внешний вид, цвета и шрифты могу отличаться от представленного здесь экземпляра, однако оси и структура такие:

Где горизонтальная ось – итерации, а вертикальная – сравниваемые параметры на протяжении моделирования. Экспортируемое разрешение изображения 1920x1080

* 1. **Описание и обоснование выбора и состава технических и программных средств**
     1. **Описание технических и программных средств**

Полностью исправный персональный компьютер под операционной системой Windows 8.1 и выше с исправными клавиатурой и компьютерной мышью. Процессор Intel или AMD с не менее чем 4 ядрами, с тактовой частотой не менее чем 1,2. Не менее 8 Гб оперативной памяти, не менее 40 Гб свободного пространства на жестком диске, где устанавливается программа. На компьютере должен быть установлен программный пакет JRE версии 14 и выше для запуска Java приложений.

* + 1. **Обоснование выбора технических и программных средств**
       1. **Язык Java**

Язык Java был выбран ввиду его кроссплатформенности, высокой производительности и широкой поддержки актуальных для этой работы библиотек, таких как GeoTools и JTS. Более того, Java обладает хорошей поддержкой объектно-ориентированного программирования, что упрощает разработку и поддержку кода.

* + - 1. **Библиотека GeoTools**

GeoTools была выбрана в качестве основной библиотеки для обработки файлов карт, так как она поддерживает множество форматов геоданных, включая форматы "shp" и "dbf". Библиотека открыто распространяется и вместе с ней распространено огромное количество примеров работы каждого из методов, обширная документация.

* + - 1. **Библиотека JTS**

JTS была выбрана в качестве основной библиотеки для создания модели клеточного автомата, так как она предоставляет набор инструментов для работы с геометрическими объектами, включая операции пересечения и объединения. Библиотека обладает высокой производительностью и хорошо документирована.

* + - 1. **Библиотека JFreeChart**

Данная библиотека была выбрана за ее простоту использования, бесплатное открытое распространение и качественную документацию С помощью библиотеки можно быстро обработать и упаковать большой массив данных в удобочитаемый график включая столбчатый.

* + - 1. **Технические средства.**

Представлена комплектация персонального компьютера для ориентировочного выполнения моделирования 17000 объектов карты не более, чем за 10 минут.

1. **ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ**

## Предполагаемая потребность

Предполагается, что программа будет использована урбанистами, инженерами и проектировщиками дорожных сетей, разработчиками геоинформационных систем.

## Ориентировочная экономическая эффективность

Данная работа не является коммерческим проектом, поэтому расчет экономической эффективности не предусмотрен.

## Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

В ходе анализа существующих решений с открытым доступом выявлено не было, поэтому проект является уникальным в своем роде.

1. **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**
2. ГОСТ 19.102-77 Стадии разработки. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.101-77 Виды программ и программных документов. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.103-77 Обозначения программ и программных документов. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.104-78 Основные надписи. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.105-78 Общие требования к программным документам. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. Cay S. Horstmann Core Java Volume II (10th edition) – Prentice Hall, 2016, 957 страниц
10. Java Topology Suite (JTS), <http://www.vividsolutions.com/jts/JTSHome.htm>
11. GeoTools The Open Source Java GIS Toolkit <https://www.geotools.org/>
12. Java-библиотека JFreeChart <https://jfree.org/jfreechart/>

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированх |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |