

МинОБРНАУКИ РОССИИ

федеральное государственное АВТОНОМНОЕ  
образовательное учреждение высшего  
профессионального образования   
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ  
УНИВЕРСИТЕТ имени академика С.П. КОРОЛЕВА   
(национальный исследовательский университет)»   
(сгау)

Факультет информатики

Кафедра информационных систем и технологий

**Отчет**

по лабораторной работе

по курсу:

«Системное программирование»

на тему:

«Автоматизированная система моделирования движения на перекрестке с круговым движением»

Выполнил: студенты группы №6302   
Мурзин А.В.,

Зарецкий А.К.

Проверил: профессор кафедры ИСТ

Михеева Т.И.

Самара, 2015

МинОБРНАУКИ РОССИИ

федеральное государственное АВТОНОМНОЕ  
образовательное учреждение высшего  
профессионального образования  
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ  
УНИВЕРСИТЕТ имени академика С. П. КОРОЛЕВА  
(национальный исследовательский университет)»  
(сгау)

Факультет информатики

Кафедра информационных систем и технологий

«УТВЕРЖДАЮ»

Профессор кафедры ИСТ

Михеева Т.И

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2015 г.

**ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ**

Студентам Мурзину Алексею Владимировичу, Зарецкому Алексею Константиновичу группы 6302

1. Тема лабораторной работы: Автоматизированная система моделирования движения на перекрестке с круговым движением
2. Исходные данные к лабораторной работе: см. приложение к заданию
3. Перечень вопросов, подлежащих разработке в лабораторной работе:
4. Системотехническая часть: обзор существующих систем, построение концепции системы, информационные и ресурсные расчеты.
5. Конструкторско-технологическая часть: разработка интерфейса системы, информационного и программного обеспечения, отладка и тестирование системы.
6. График выполнения лабораторной работы:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Этапы лабораторной работы | % | Сроки выполнения  по этапам | Итоги проверки | | |
|  |  |  |
| 1. Обзор литературы, изучение предметной области | 22 | 11.03.2015 |  |  |  |
| 2. Разработка информационной модели | 16 | 25.03.2015 |  |  |  |
| 3. Разработка функциональной схемы системы | 28 | 20.04.2015 |  |  |  |
| 4. Разработка алгоритмов | 3 | 22.04.2015 |  |  |  |
| 6. Отладка программы | 23 | 13.05.2015 |  |  |  |
| 7. Написание пояснительной записки | 8 | 20.05.2015 |  |  |  |

1. Срок представления на кафедру законченного проекта: 20 мая 2015 г.
2. Дата выдачи задания: 19 февраля 2015 г.

*Руководитель лабораторной работы*  Михеева Т. И., профессор каф. ИСТ

(фамилия, и., о., должность, подпись)

*Задание принял к исполнению 20 февраля 2015 г*

(дата)

**ПРИЛОЖЕНИЕ**

**к заданию на лабораторную работу по теме:**«Автоматизированная система моделирования движения на перекрестке с круговым движением»

**студентов группы №6302 Мурзина Алексея Владимировича и Зарецкого Алексея Константиновича**

Исходные данные к проекту:

1. Характеристики объекта автоматизации:
2. Вид автоматизируемой деятельности: процесс моделирования движения на перекрестке с круговым движением.
3. Количество пересекаемых дорог перекрестка: 2
4. Количество полос для движения в каждом направлении: 1-4
5. Наличие пешеходного перехода перед перекрестком: всегда
6. Наличие светофора регулирующий, пешеходный переход: опционально.
7. Виды приоритетов движения:
   * Главное кольцо
   * Второстепенное кольцо
   * Главная одна из улиц
8. Требования к информационному обеспечению:
9. Информационное обеспечение разрабатывается на основе следующих документов:
   * Постановление Правительства РФ от 23.10.1993 N 1090 (ред. от 20.04.2015) "О Правилах дорожного движения" [1]
10. Визуализация на экране перекрестка, кольца, дорожных знаков, сигнала светофора, движения машин и пешеходов.
11. Хранение созданных моделей перекрестков в базе данных.
12. Предусмотреть ограничения целостности при вводе и редактировании данных.
13. Требования к техническому обеспечению:
14. Тип ЭВМ – IBM PC совместимый.
15. Конфигурация комплекса технических средств определяется в процессе выполнения лабораторной работы.
16. Требования к программному обеспечению:
17. Тип операционной системы – MS Windows XP и выше, платформа x86 и x64
18. Среда программирования – MS Visual Studio 2013.
19. Язык программирования – C#, платформа Microsoft .NET Framework 4.5
20. Тип используемой СУБД – SQLite 3.8.9
21. Инструментальные средства разработки – определяются в процессе выполнения лабораторной работы.
22. Общие требования к проектируемой системе.
23. Функции, реализуемые системой:
24. Выполнение разграничений прав доступа (суперпользователь, расширенный пользователь, пользователь).
25. Управление моделями перекрестков для пользователей с правами суперпользователя: создание, изменение, удаление.
26. Управление пользователями для пользователя с правами суперпользователя: добавление, изменение уровня прав, удаление.
27. Изменения закона распределения машин и пешеходов во время моделирования для пользователя с правами не ниже расширенного пользователя.
28. Технические требования к системе:
    * режим работы – диалоговый;
    * время реакции при вводе/выводе данных в систему – не более 10 секунд;
    * система должна удовлетворять санитарным правилам и нормам СанПин 2.2.2/2.4 2198-08;
    * условия работы средств вычислительной техники должны соответствовать группе 1 пункта 1.3.1 ГОСТ 12.1.005, 12.1.007.
29. Условия труда:

* температура окружающего воздуха – 15-25 оС;
* относительная влажность – 45-75%;
* содержание вредных веществ, пыли и подвижного воздуха в рабочей зоне соответствует нормам ГОСТ 12.1.005, 12.1.007.

Руководитель лабораторной работы Михеева Т. И.

Задание приняли к исполнению Мурзин А.В.

Зарецкий А.К.

**РЕФЕРАТ**

Пояснительная записка: 44 страницы, 21 рисунок, 7 таблиц, 12 источников, 2 приложения.

ПЕРЕКРЕСТОК, КРУГОВОЕ ДОРОЖНОЕ ДВИЖЕНИЕ, ИММИТАЦИОННОЕ МОДЕЛИРОВАНИЕ, АВТОМАТИЗИРОВАННАЯ СИСТЕМА МОДЕЛИРОВАНИЯ ДОРОЖНОГО ДВИЖЕНИЯ

Созданная программа, названная RoadRingSim, является средой для имитационного моделирования дорожного движения на перекрестке с круговым движением.

Реализованная модель представляет объект исследования с достаточной точностью. В модели приняты следующие упрощения:

* Все автомобили имеют одинаковые динамические характеристики и габариты
* У всех автомобилей одинаковая модель поведения и у всех пешеходов
* Не учитывается ускорение автомобилей и пешеходов, связанные с их инертностью
* Дорожное полотно разбито на квадраты, в каждом из которых в каждую единицу модельного времени может находиться ровно одна машина или пешеход и полностью занимать его

Оглавление

[ВВЕДЕНИЕ 10](#_Toc419848143)

[1 СИСТЕМОТЕХНИЧЕСКАЯ ЧАСТЬ 11](#_Toc419848144)

[1.1 Описание и анализ предметной области 11](#_Toc419848145)

[1.1.1 Предметная область 11](#_Toc419848146)

[1.2 Обзор систем-аналогов 12](#_Toc419848147)

[1.2.1 Аналог 2 12](#_Toc419848148)

[1.2.2 Аналог 2 12](#_Toc419848149)

[1.2.3 Аналог 3 12](#_Toc419848150)

[1.3 Описание логической структуры системы 13](#_Toc419848151)

[1.4 Описание логической модели базы данных 14](#_Toc419848152)

[1.5 Выбор и обоснование комплекса программных средств 15](#_Toc419848153)

[1.5.1 Выбор операционной системы 15](#_Toc419848154)

[1.5.2 Выбор языка программирования 16](#_Toc419848155)

[1.5.3 Выбор среды разработки 16](#_Toc419848156)

[1.5.4 Выбор системы управления базами данных 16](#_Toc419848157)

[1.6 Выбор и обоснование комплекса технических средств 17](#_Toc419848158)

[1.6.1 Расчет емкости ОЗУ 17](#_Toc419848159)

[1.6.2 Расчет емкости дискового пространства 18](#_Toc419848160)

[1.6.3 Расчет времени реакции системы 19](#_Toc419848161)

[1.6.4 Минимальные и рекомендованные характеристики технических средств 19](#_Toc419848162)

[2 КОНСТРУКТОРСКО-ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 20](#_Toc419848163)

[2.1 Архитектура системы 20](#_Toc419848164)

[2.1.1 Технологии построения системы 20](#_Toc419848165)

[2.1.2 Паттерны проектирования 20](#_Toc419848166)

[2.1.2.1 Паттерн абстрактная фабрика 20](#_Toc419848167)

[2.1.2.2 Паттерн одиночка (Singleton) 21](#_Toc419848168)

[2.1.2.3 Паттерн машина состояний (State Machine) 22](#_Toc419848169)

[2.1.3 Описание физической модели данных 23](#_Toc419848170)

[2.1.4 Описание алгоритмов функционирования системы 27](#_Toc419848171)

[2.1.5 Описание программных модулей системы 29](#_Toc419848172)

[ЗАКЛЮЧЕНИЕ 30](#_Toc419848173)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 31](#_Toc419848174)

[ПРИЛОЖЕНИЕ А Листинг программы 32](#_Toc419848175)

[ПРИЛОЖЕНИЕ Б Руководство пользователя 40](#_Toc419848176)

**ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

АСИ – автоматизированная система моделирования

ПДД – правила дорожного движения

ПрО – предметная область;

ОС – операционная система;

СУБД – система управления базами данных;

LINQ – Language Integrated Query;

IDE – Integrated development environment;

БД – база данных;

БЖД – безопасность жизнедеятельности;

КСА – комплекс средств автоматизации;

КТС – комплекс технических средств;

ОЗУ – оперативное запоминающее устройство;

ПО – программное обеспечение;

ЭВМ – электронно-вычислительная машина;

DAO – Data Access Object;

ER – Entity – Relationship

CASE – Computer-Aided Software Engineering

MVP – Model View Presenter;

SQL – Structured Query Language;

UML – Unified Modeling Language;

API – application programming interface;

ВВЕДЕНИЕ

Объектом исследования настоящей работы является анализ ситуации на дороге в условиях различного количества автомобильного и пешеходного трафика. Предмет исследования – перекресток двух дорог с круговым движением.

Разработанная программа решает проблему определения параметров предмета исследования, исходя из заданных условий. Такими параметрами являются: количество полос дорожного полотна, приоритеты движения при въезде на кольцо, регламентируемых в ПДД, наличие светофора, регулирующего движение на пешеходном переходе, режим его работы.

В настоящее время в нашей стране, и в частности в нашем городе, имеется большое количество изношенного дорожного полотна, растет количество дорожного трафика. Сложившаяся ситуация требует все больше расходов на ремонт, модернизацию и расширение дорог. Поэтому проблема определения параметров требуемых работ с целью оптимизации экономических расходов сегодня очень актуальна.

Экспериментировать на реальном объекте исследования сложно и дорого. Аналитическое моделирование нецелесообразно или даже невозможно, так как объект исследования содержит причинно-следственные связи, связанные со сложным поведением машин и пешеходов, стохастические явления, требует учет поведения системы во времени.

Очевидным, наглядным и гибким способом решения проблемы является разработка симулятора предмета исследования для проведения экспериментов над ним. В этом и состоит цель настоящей работы.

Для достижения цели были поставлены и решены следующие задачи

* Ознакомление с предметной областью и программами-аналогами
* Выбор средств и инструментов для разработки
* Разработка автоматизированной системы моделирования движения на перекрестке с круговым движением, далее просто система или программа RoadRingSim

1. СИСТЕМОТЕХНИЧЕСКАЯ ЧАСТЬ
   1. Описание и анализ предметной области

Проектирование программной системы начинается с описания и анализа ее предметной области (ПрО), принципов работы и требований, которым она должна будет удовлетворять.

Предметная область представляется множеством фрагментов – элементов реального мира, подлежащий изучению с целью организации управления и, в конечном счете, автоматизации. Каждый фрагмент предметной области характеризуется множеством объектов и процессов, использующих объекты, а также множеством пользователей, характеризуемых различными взглядами на предметную область.

* + 1. Предметная область

Дорожное полотно или просто «дорога» характеризуется количеством полос для движения автомобилей в одном направлении. Полоса движения - любая из продольных полос проезжей части, обозначенная или не обозначенная разметкой и имеющая ширину, достаточную для движения автомобилей в один ряд [1].

Характеристикой автомобильного (пешеходного) трафика является время между появлением новых автомобилей (пешеходов) на исследуемом участке дороги. Это время описывается детерминированным или случайной величиной с заданным законом распределения вероятностей. Далее эту характеристику будем называть «законом распределения автомобилей (пешеходов)» или просто «распределение автомобилей (пешеходов)»

Очередность проезда перекрестков устанавливают знаки приоритета:

"Уступите дорогу". Водитель должен уступить дорогу транспортным средствам, движущимся по пересекаемой дороге.



Рисунок 1.1 – Знак «Уступите дорогу»

"Главная дорога". Дорога, на которой предоставлено право преимущественного проезда нерегулируемых перекрестков [1]

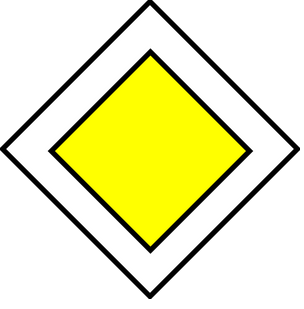


Рисунок 1.2 – Знак «Главная дорога»

* 1. Обзор систем-аналогов

Описание аналога

* + 1. Aimsun

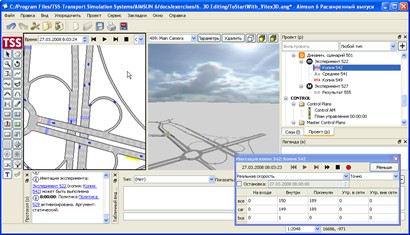
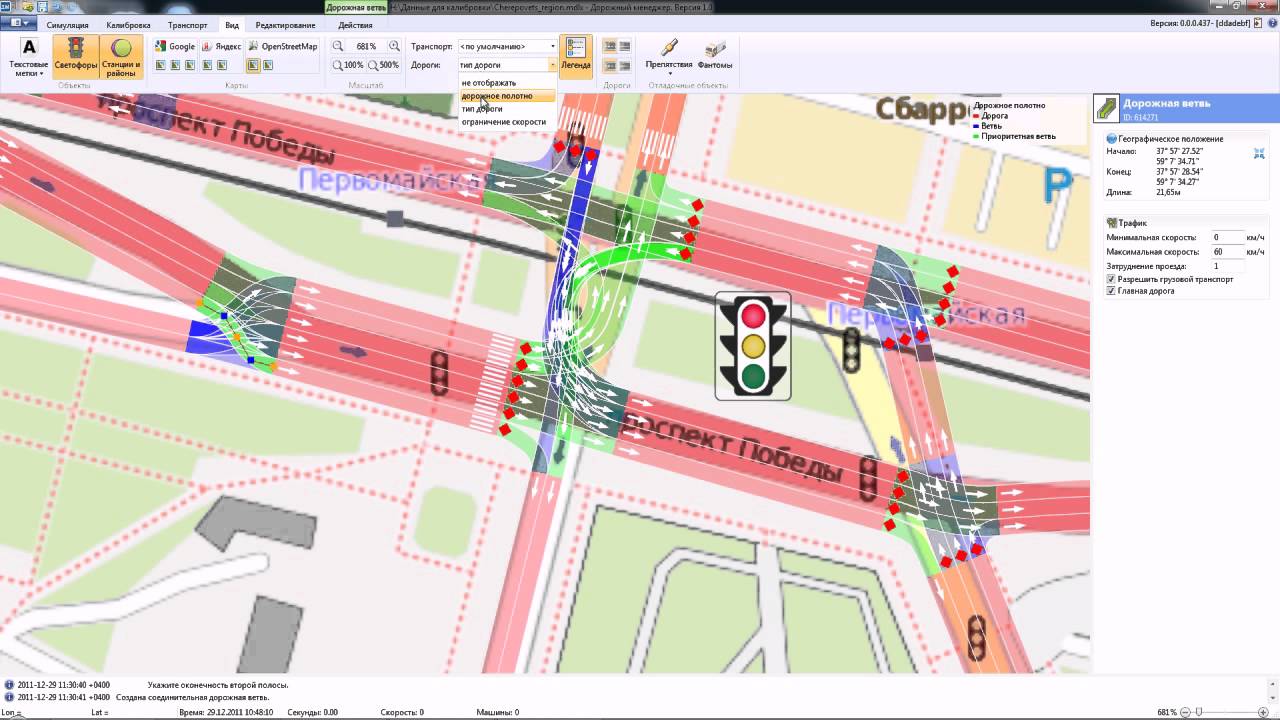


Рисунок 1.3 – скриншот программы Aimsun

Представляет собой полнофункциональный комплекс инструментов анализа транспортных потоков и перевозок, который может использоваться для планирования, детального моделирования и исследования требований и условий деятельности в сфере транспорта. Продукт реализует интегрированную платформу, пригодную для выполнения как статического, так и динамического моделирования. Представляет собой расширяемую программную среду, архитектура которой допускает наличие неограниченного количества компонентов, реализованных TSS, сторонними разработчиками или самими пользователями. Пользователь обладает теми же полными возможностями программируемого доступа к инструментам AIMSUN и средствами их адаптации, что и создатели системы.

* + 1. Дорожный менеджер

Рисунок 1.4 – скриншот программы Дорожный менеджер

Предназначен для моделирования транспортных потоков в улично-дорожной сети города или региона. Позволяет наглядно увидеть обстановку на улично-дорожной сети в случае различных дорожно-транспортных ситуаций или при изменении конфигурации сети. Для построения модели транспортных потоков в программе не требуется вносить большое количество статистических данных о населении, рабочих местах, образовательных учреждениях и т.д. Формирование модели осуществляется на основе эмпирических данных по потокам на различных участках УДС и информации о свойствах этих участков. Применяемый подход позволяет получить высокую степень адекватности транспортных потоков в модели. Алгоритм движения транспортных средств основан на принципах выбора маршрутов реальными водителями и особенностях их езды. При этом все параметры данного алгоритма могут быть изменены пользователем с целью моделирования различных вариантов поведения водителей. В программе реализовано имитационное моделирование сложных динамических систем на основе дискретно-событийного подхода, что дает возможность выполнить симуляцию движения и взаимодействия десятков тысяч транспортных средств на УДС. Для построения модели УДС не требуется наличия ГИС-информации. Модель улично-дорожной сети не имеет территориальных ограничений: возможно построение модели как для отдельного района города, так и для целого региона. При работе с программой не требуются профессиональные знания в области транспортного моделирования. Программа имеет современный интерфейс, легкий для понимания и освоения.

* + 1. Visum

Представляет собой обширную, гибкую программу для моделирования транспортных потоков, расчета спроса на транспорт (матрицы корреспонденций для общественного и индивидуального транспорта), анализа транспортной сети, расчет себестоимости общественного транспорта и прогноза запланированных мероприятий и их последствий. Используется для моделирования транспортных потоков, транспортного планирования и оптимизации общественного транспорта: в городах, регионах, мегаполисах. VISUM интегрирует всех участников движения в единую математическую транспортную модель.

* 1. Описание логической структуры системы

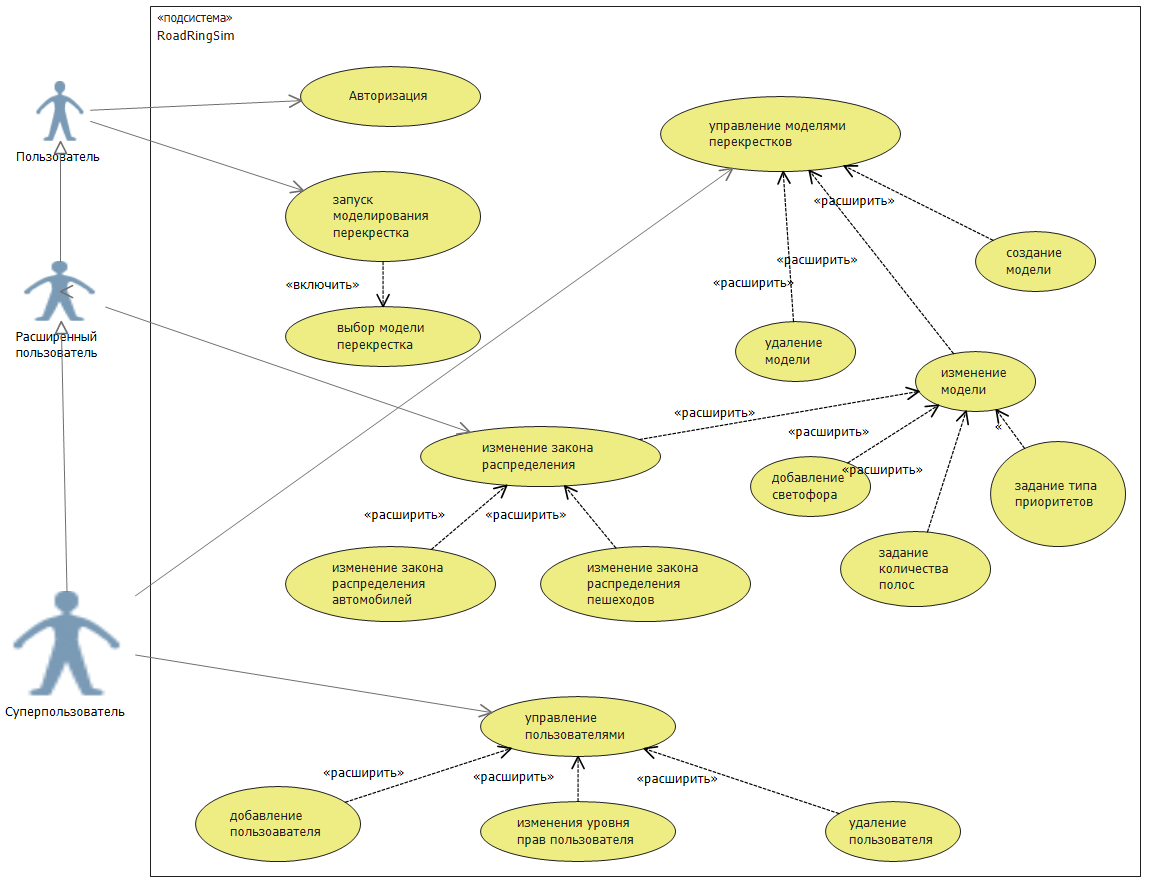


Рисунок 1.5 - Диаграмма вариантов использования

* 1. Описание логической модели базы данных

Логические модели данных универсальны. Посредством этих моделей отображается абстрактный взгляд на данные. Объекты логической модели соответствуют объектам реального мира и могут носить такие же названия.

Логическая модель независима и не связана с конкретной СУБД. Сущности в логической модели описываются набором атрибутов, но при этом не указывается тип данных атрибутов. Реализация логической модели в среде конкретной СУБД будет оригинальной для каждой СУБД. Поэтому одной логической модели могут соответствовать несколько физических моделей. Для каждой СУБД – своя физическая модель.

Основными компонентами логической модели являются сущности, атрибуты и связи.

В качестве сущности может выступать объект, событие, процесс или концепция. Каждый атрибут является характеристикой сущности, описанием ее свойства. Связи показывают, как сущности соотносятся друг с другом логически. Отношение числа экземпляров родительской сущности к числу экземпляров дочерней отображается при помощи Мощности связи.

Генерация физической схемы базы данных из логической модели данных выполняется называется прямым проектированием. Возможна обратная генерация логической модели данных из физической схемы. Этот процесс называется обратным проектированием. [2]

Общим способом представления логической модели БД является построение ER-диаграмм (Entity – Relationship, сущность - связь).

Методология IDEF1X служит для описания модели данных на языке и в терминах реляционной модели данных. CASE - средства (инструментальные средства автоматизированного проектирования баз данных), в частности ERwin, поддерживающие эту методологию, позволяют строить логическую, независимую от СУБД модель данных для общего представления системы и входящих в нее объектов и физическую модель данных, которая может быть трансформирована в любую реляционную СУБД и описана на языке описания данных этой СУБД [2]

Для построения ER – диаграммы по методологии IDEF1X воспользуемся CASE - средством Erwin

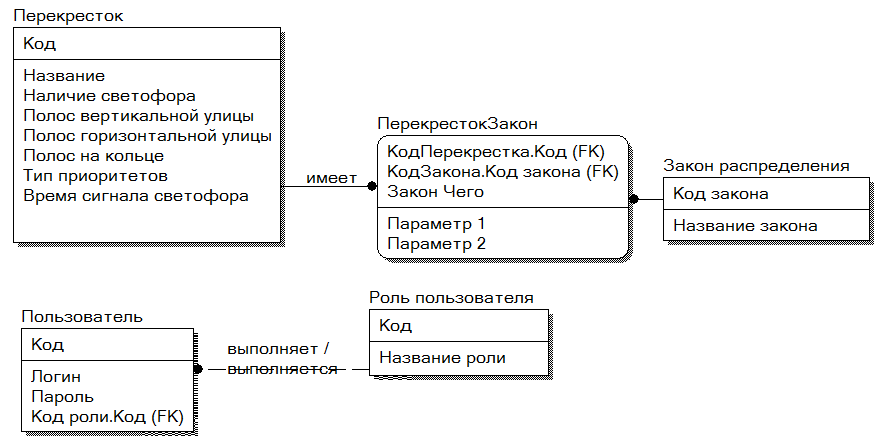


Рисунок 1.6 - Логическая модель базы данных (ER-модель), выполненная по методологии IDEF1x.

* 1. Выбор и обоснование комплекса программных средств
     1. Выбор операционной системы

Выбрана ОС Microsoft Windows 8.1, предназначенная для персональных компьютеров. На момент разработки является актуальной версией ОС семейства Windows NT. [3]

Основной причиной выбора ОС семейства Windows NT стало наличие эффективной и современной интегрированной среды разработки Microsoft Visual Studio 2013, позволяющей разрабатывать приложения с графически интерфейсом для платформы .Net Framework 4.5.

Windows – самая распространенная ОС на территории Российской Федерации и в мире. По состоянию на август 2014 года под управлением операционных систем семейства Windows работает более 91% персональных компьютеров [4]

* + 1. Выбор языка программирования

Выбран язык объектно-ориентированный язык программирования С#, как язык для разработки приложения для платформы Microsoft .NET Framework 4.5.

Выбор был обоснован тем, что этот язык имеет простой и элегантный С-подобный синтаксис, поддержку работы с СУБД, механизм LINQ-запросов, полный доступ к Base Class Library - стандартной библиотеке классов платформы .NET. Для разработки приложений на С# имеется удобная IDE Visual Studio.

* + 1. Выбор среды разработки

Так как в качестве языка программирования был выбран C#, то наиболее очевидным выбором является MS Visual Studio 2013. Это очень удобная и бесплатная IDE для разработки ПО. MS Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью рефакторинга и отладки кода.

Встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, редактор UML – диаграмм. Среди полезных расширений Visual Studio хочется отметить NuGet – менеджер для управления зависимостями от сторонних библиотек. С помощью этого инструмента можно устанавливать, обновлять и убирать зависимости для вашего проекта с большой легкостью [5]

* + 1. Выбор системы управления базами данных

Так как приложение не является большим, то целесообразен выбор компактной встраиваемой СУБД SQLite. [Движок](http://ru.wikipedia.org/wiki/%D0%94%D0%B2%D0%B8%D0%B6%D0%BE%D0%BA) SQLite предоставляет [библиотеку](http://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), с которой программа компонуется и он становится составной частью программы, а в качестве протокола обмена используются вызовы функций ([API](http://ru.wikipedia.org/wiki/API)) этой библиотеки. [6]

SQLite – это, пожалуй, самая распространенная встраиваемая в приложение легковесная реляционная СУБД, поддерживающая подмножество SQL. Она очень надежная: в тестировании SQLite используется три набора тестов, покрытие кода тестами 100%. [7]

* 1. Выбор и обоснование комплекса технических средств
     1. Расчет емкости ОЗУ

Для расчета объема ОЗУ, необходимого для нормальной работы системы, воспользуемся формулой:

,

где  – объем оперативной памяти, необходимый для нормальной работы операционной системы,

 – объем оперативной памяти, необходимый для нормальной работы программы,

 – объем кэша для хранения данных, загружаемых в оперативную память при работе системы,

 – объем памяти, используемой системой управления базами данных.

Расчет будем проводить для выбранной ОС Windows 8.1 64-разрядная.

 = 2048 Мб. [8]

Согласно формуле, объем памяти, необходимый для хранения программ определяется объемом памяти, который занимает программа.

Для хранения системы необходимо 6,4 Мб.

Таким образом, получаем, что:

 = 10,7 Мб.

Опытным путем установлено, что для хранения атрибутивных данных в памяти системе требуется 60 Мб.

 = 60 Мб.

Так как СУБД SQLite является встраиваемой, то расходы на оперативную память уже включены в .

 = 0 Мб.

Таким образом, общий объем ОЗУ составляет

 = 2048 + 10,7 + 60 + 0 = 2118,7 Мб.

* + 1. Расчет емкости дискового пространства

Для расчета емкости дискового пространства, необходимого для нормальной работы системы, воспользуемся формулой:

,

где  – объем дискового пространства, необходимого для хранения файлов операционной системы с учетом рота [8]

 = 20480 Мб,

 – объем дискового пространства, необходимого для хранения файлов программы

 = 3.3 Мб,

 – объем дискового пространства, необходимого для хранения данных с учетом роста базы данных.  = 1 Мб,

 – объем дискового пространства, необходимого для хранения файлов СУБД = 12.5 Мб.

Таким образом, общий объем дискового пространства составляет:

 = 20480+3,3+1+12,5 = 20496,8 Мб.

* + 1. Расчет времени реакции системы

Для расчета времени реакции системы воспользуемся формулой:

,

где  – время, затрачиваемое процессором на выполнение команд;

 – время, затрачиваемое при обращении к жесткому диску для считывания блоков данных;

 – время, затрачиваемое системой на вывод информации на экран монитора.

Так как время реакции не является критическим параметром для данной системы, то его определение проводилось опытным путем. Результаты показали, что система отвечает требованиям, указанным в задании при частоте процессора 3 ГГц.

* + 1. Минимальные и рекомендованные характеристики технических средств

Минимальные требования к техническим характеристикам и функционированию рабочей станции:

* операционная система: Windows XP и новее;
* процессор – Intel Pentium не менее 1,5 ГГц (от 800 ГГц);
* объем оперативной памяти – 1 Гб (от 450Мб);
* свободное место на диске – 10 Гб (от 2Гб);
* сетевой адаптер – 54/100 Мбит;
* рабочая станция должна иметь доступ в интернет;
* рабочая станция должна быть включена постоянно в течение рабочего времени.

1. КОНСТРУКТОРСКО-ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ
   1. Архитектура системы

В нашей программе используется встраиваемая СУБД SQLite. Слово «встраиваемый» означает, что SQLite не использует парадигму клиент-сервер, то есть движок SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а предоставляет библиотеку, с которой программа компонуется и движок становится составной частью программы. Таким образом, в качестве протокола обмена используются вызовы функций (API) библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу. SQLite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором исполняется программа. Простота реализации достигается за счёт того, что перед началом исполнения транзакции записи весь файл, хранящий базу данных, блокируется. [6]

* + 1. Технологии построения системы
    2. Паттерны проектирования

Любой паттерн проектирования, используемый при разработке информационных систем, представляет собой формализованное описание часто встречающейся задачи проектирования, удачное решение данной задачи, а также рекомендации по применению этого решения в различных ситуациях. [9]

* + - 1. Паттерн абстрактная фабрика

Абстрактная фабрика предоставляет интерфейс, позволяющий порождать семейства объектов c заданными интерфейсами. При этом их реализации могут варьироваться. Такой подход уменьшает зависимости между объектами и повышает гибкость, за счет возможности изменения реализаций. [10]

С использованием этого паттерна создан класс, описывающий случайную величину, распределённую по заданному закону распределения. Алгоритм метода GetSample() зависит от поля Type, которое может принимать одно из значений, заданных в перечислении DistributionLaws.

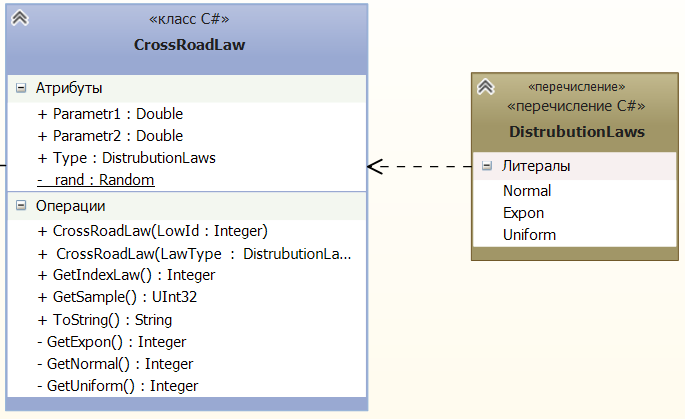


Рисунок 2.1 – UML диаграмма реализации паттерна «Абстрактная фабрика»

* + - 1. Паттерн одиночка (Singleton)

Проблема: Необходим лишь один экземпляр специального класса, различные объекты должны обращаться к этому экземпляру через единственную точку доступа.

Решение: Создать класс и определить статический метод класса, возвращающий этот единственный объект. [9]

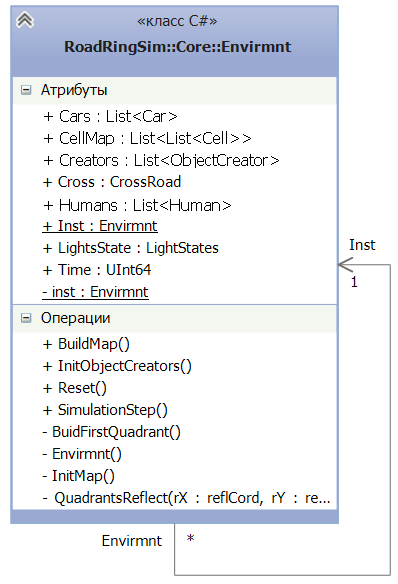


Рисунок 2.2 – UML диаграмма реализации паттерна «Одиночка»

Класс Envirmnt описывает объекты среды, в которой происходит моделирование. Так как в каждый момент времени происходит моделирование только одной модели перекрестка, то требуется только один экземпляр этого класса. Статическое свойство Inst проверяет, создан ли экземпляр этого класса, и создает его, сохраняя ссылку в поле inst, если это не так. Метод Reset позволяет удалить ссылку на созданный экземпляр, если требуется создание нового, например, при полном перезапуске среды моделирования.

* + - 1. Паттерн машина состояний (State Machine)

Это поведенческий паттерн, который является объектно-ориентированной реализацией конечного автомата.

Решаемая проблема: поведение объекта зависит от его состояния и должно изменяться во время выполнения программы. Такую схему можно реализовать, применив множество условных операторов: на основе анализа текущего состояния объекта предпринимаются определенные действия. [11]

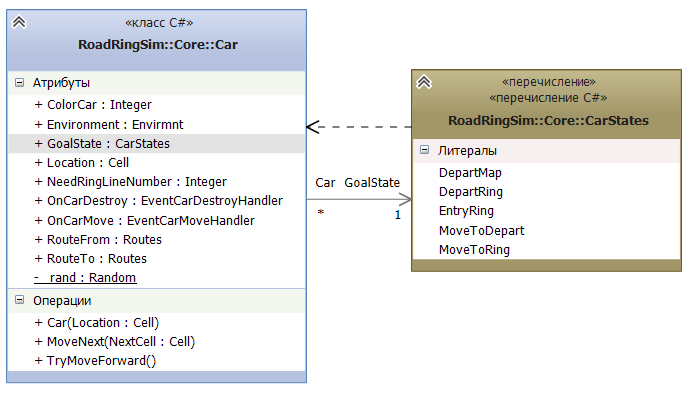


Рисунок 2.3 – UML диаграмма реализации паттерна «Машина состояний»

С помощью этого паттерна описывается алгоритм поведения автомобилей в классе Car. Метод TryMoveForward() описывает поведение автомобиля: в зависимости от значения GoalState логика поведения автомобиля разная.

* + 1. Описание физической модели данных

На более низком уровне, относительно логического, находится физическая модель данных. Физическая модель данных описывает данные средствами конкретной СУБД. Физическая модель данных реализована средствами именно реляционной СУБД. Отношения, разработанные на стадии формирования логической модели данных, преобразуются в таблицы, атрибуты становятся столбцами таблиц, для ключевых атрибутов создаются уникальные индексы, домены преображаются в типы данных, принятые в конкретной СУБД.

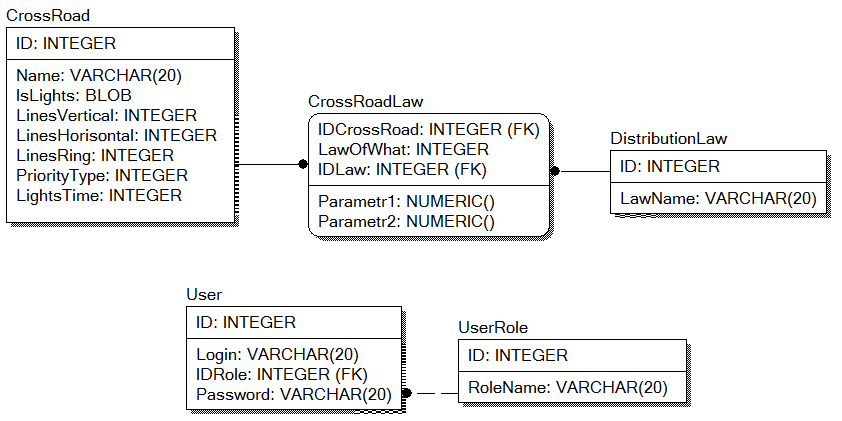


Рисунок 2.4 – Физическая модель базы данных

Таблица 2.2 – Соответствие сущностей логической и физической моделей

|  |  |
| --- | --- |
| Сущность | Таблица |
| Перекресток | CrossRoad |
| Закон перекрестка | CrossRoadLaw |
| Закон распределения | DistributionLaw |
| Пользователь | User |
| Роль пользователя | UserRole |

Таблица 2.3 – Таблица CrossRoad

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Имя столбца** | **Тип** | **Размер (байт)** |
| **Код** | **ID** | **INTEGER** | **4** |
| Название | Name | VARCHAR(20) | 22 |
| Наличие светофора | IsLights | BLOB | 1 |
| Полос вертикальной улицы | LinesVertical | INTEGER | 4 |
| Полос горизонтальной улицы | LinesHorisontal | INTEGER | 4 |
| Полос на кольце | LinesRing | INTEGER | 4 |
| Тип приоритетов | PriorityType | INTEGER | 4 |
| Время сигнала светофора | LightsTime | INTEGER | 4 |
| Итого |  |  | 47 |

Таблица 2.4 – Таблица CrossRoadLaw

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Имя столбца** | **Тип** | **Размер (байт)** |
| **Код перекрестка** | **IDCrossRoad** | **INTEGER** | **4** |
| **Код закона** | **IDLaw** | **INTEGER** | **4** |
| **Закон чего** | **LawOfWhat** | **INTEGER** | **4** |
| Параметр 1 | Parametr1 | NUMERIC | 8 |
| Параметр 2 | Parametr2 | NUMERIC | 8 |
| Итого |  |  | 28 |

Таблица 2.5 – Таблица DistributionLaw

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Имя столбца** | **Тип** | **Размер (байт)** |
| **Код закона** | **ID** | **INTEGER** | **4** |
| Название закона | Name | VARCHAR(20) | 22 |
| Итого |  |  | 26 |

Таблица 2.6 – Таблица User

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Имя столбца** | **Тип** | **Размер (байт)** |
| **Код** | **ID** | **INTEGER** | **4** |
| Логин | Login | VARCHAR(20) | 22 |
| Пароль | Password | VARCHAR(20) | 22 |
| Код роли | IDRole | INTEGER | 4 |
| Итого |  |  | 52 |

Таблица 2.7 – Таблица UserRole

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Имя столбца** | **Тип** | **Размер (байт)** |
| **Код** | **ID** | **INTEGER** | **4** |
| Название роли | RoleName | VARCHAR(20) | 22 |
| Итого |  |  | 26 |

* + 1. Описание алгоритмов функционирования системы

Алгоритм – это точный набор инструкций, описывающих порядок действий некоторого исполнителя для достижения результата, решения некоторой задачи за конечное время.

На рисунке 2.6 представлена схема главного алгоритма моделирования. Перед пояснением алгоритма рассмотрим классы среды моделирования, рисунок 2.5. В основном классе среды моделирования Envirmnt имеется список объектов типа ObjectCreator, который представляет собой абстрактный класс Создателей объектов (автомобилей типа Car, пешеходов типа Human) в среде моделирования. Конкретные реализации этого класса CarCreator и HumanCreator описывают логику создания и планирования соответственно автомобилей и пешеходов. Класс Envirmnt содержит список всех создателей объектов типа ObjectCreator, объектов автомобилей типа Car и объектов пешеходов типа Human.

Алгоритм, схема которого представлена на рисунке 2.6, описывает метод SimulationStep() класса Envirmnt, который многократно вызывается в процессе моделирования. В этом алгоритме для всех элементов выше упомянутых списков последовательно вызываются методы создания новых объектов, инкапсулированные в экземпляры класса ObjectCreator, методы TryMoveForward(), описывающие алгоритмы поведения автомобилей и пешеходов, инкапсулированные в классе Car и Human.

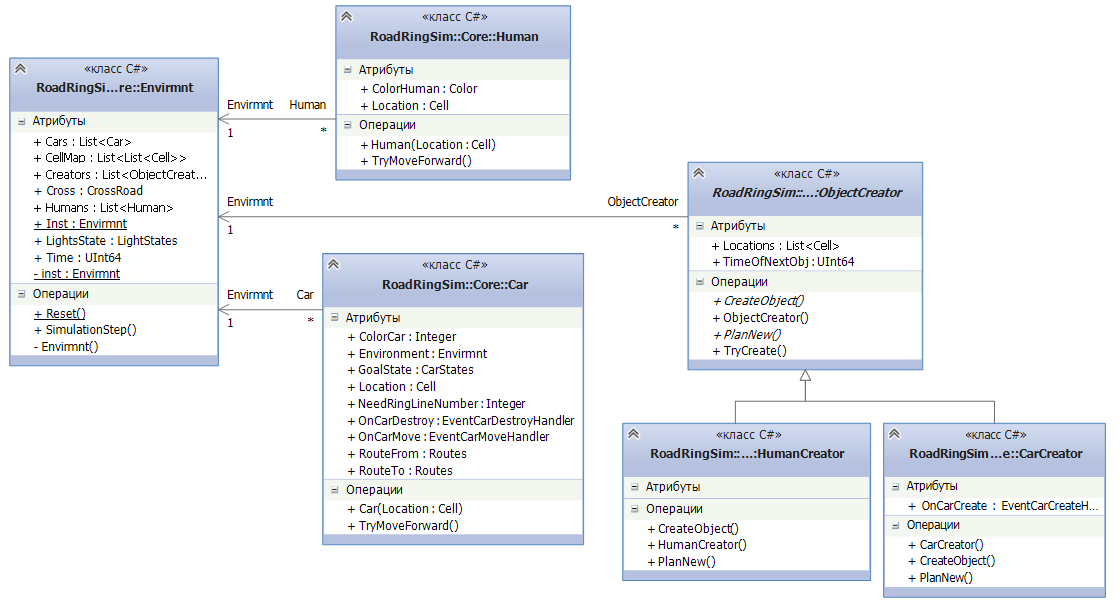


Рисунок 2.5 – UML диаграмма классов среды моделирования



Рисунок 2.6 - Схема главного алгоритма моделирования

* + 1. Описание программных модулей системы

Автоматизированная система состоит из следующих основных программных модулей:

* RoadRing.Data – модуль системы, в котором описано взаимодействие с базой данных, используется паттерн DAO.

Паттерн DAO предназначен для отделения взаимодействия с хранилищем данных от бизнес-логики приложения. Т.е. вся логика, отвечающая за сохранение, изменение, извлечение сущностей выносится в отдельные DAO-классы, а код, инкапсулирующий бизнес-логику приложения, взаимодействует с этими классами, а не непосредственно с хранилищем данных. Такой подход обеспечивает гибкость в выборе подсистемы хранения для приложения. [12]

* RoadRing.Core – модуль системы, в котором описаны классы бизнес логики приложения: классы симулятора дорожного движения, сущностей базы данных (пользователи, модели перекрестков).
* RoadRing.Model, RoadRing.Forms, RoadRing.Presenter реализуют графический интерфейс пользователя с использованием паттерна MVP.
* RoadRing.Render содержит описание логики визуализации среды моделирования.

ЗАКЛЮЧЕНИЕ

Была достигнута цель работы – разработан симулятор дорожного движения на перекрестке с круговым движением. Была получена полноценная функциональная автоматизированная система моделирования, соответствующая требованиям к информационному, техническому, программному обеспечению, а также требованиям к проектируемой системе. Разработаны информационная и функциональная модели системы, написаны алгоритмы, отлажена создана программа. Изучены аналоги продукта и выбраны, в результате исследований, средства разработки и комплекс используемых технических средств.

Использование созданной программы позволило оптимальные определить параметры перекрестков при заданных законах распределения автомобилей и пешеходов при которых будет обеспечена достаточная пропускная способность. Результатом моделирования при заданных параметрах является факт возникновения взаимной блокировки участников движения (тупика) или функционирование монитора моделирования сколь угодно долгое время.

Практическая значимость результата состоит в том, что система позволяет определить оптимальный план работ при строительстве новых дорожных перекрестков с круговым движением или расширением существующих перекрестков. Для этого перед использованием системы моделирования необходимо провести статистический анализ распределения автомобильного и пешеходного трафика.

Слабым местом настоящей работы является выбор очень узкого объекта исследования. Модель предусматривает перекрестки с пересечением только двух дорог, только с одним пешеходным переходом и не предусматривает ничего, кроме перекрестков с круговым движением. В этом плане аналоги имеют явное преимущество по сравнению с разработанным решением. С другой стороны, программа получилась простой в использовании и технической поддержке.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

|  |  |
| --- | --- |
| [1] | Постановление Правительства РФ от 23.10.1993 N 1090 (ред. от 20.04.2015) "О Правилах дорожного движения", Москва, 2015. |
| [2] | Т. Васючкова и Н. Иванчева, «Технология создания информационных систем. Бизнес - моделирование. Часть2,» 2011. [В Интернете]. Available: http://mmc2.nsu.ru/default.aspx?db=book\_vasyuchkova&int=VIEW&el=41&templ=I206. |
| [3] | Windows 8.1 : Материал из Википедии — свободной энциклопедии : Версия 70247977, сохранённая в 12:20 UTC 22 апреля 2015 // Википедия, свободная энциклопедия. — Электрон. дан. — Сан-Франциско: Фонд Викимедиа, 2015. — Режим доступа: http://ru.wikipedia.org/?. |
| [4] | Windows // Википедия. [2015—2015]. Дата обновления: 13.05.2015. URL: http://ru.wikipedia.org/?oldid=70771652 (дата обращения: 13.05.2015).. |
| [5] | пользователь с псевдонимом koissakh, «NuGet использование,» 25 Апрель 2011. [В Интернете]. Available: http://softblog.violet-tape.ru/2011/04/25/nuget-simple-using/. |
| [6] | SQLite : Материал из Википедии — свободной энциклопедии : Версия 70450774, сохранённая в 13:19 UTC 27 апреля 2015 // Википедия, свободная энциклопедия. — Электрон. дан. — Сан-Франциско: Фонд Викимедиа, 2015. — Режим доступа: http://ru.wikipedia.org/?oldid. |
| [7] | Y. Akhotnikau, «Ознакомился с описанием тестирования SQLite,» 28 Декарбрь 2011. [В Интернете]. Available: http://eao197.blogspot.ru/2011/12/progtesting-sqlite.html?spref=tw. |
| [8] | «Требования к системе Windows 8.1,» [В Интернете]. Available: http://windows.microsoft.com/ru-ru/windows-8/system-requirements. |
| [9] | Д. Ольга, «Обзор паттернов проектирования,» МГУ. |
| [10] | В. Андрей, «Порождающие шаблоны: Абстрактная фабрика (Abstract factory),» 5 Октябрь 2010. [В Интернете]. Available: http://andrey.moveax.ru/post/patterns-oop-creational-abstract-factory. |
| [11] | Автор сайта cpp-refrence.ru, «Паттерн State (состояние),» [В Интернете]. Available: http://cpp-reference.ru/patterns/behavioral-patterns/state/. [Дата обращения: 19 Май 2015]. |
| [12] | П. Самолысов, «О спорном паттерне DAO,» 21 Июль 2014. [В Интернете]. Available: http://samolisov.blogspot.ru/2014/07/dao.html. |

ПРИЛОЖЕНИЕ А  
Листинг программы

namespace RoadRingSim.Core.Domains

{

public class CrossRoad : DomainObject

{

/// <summary>

/// есть ли светофор на пешеходном переходе внизу

/// </summary>

public bool IsLights { get; set; }

public CrossRoadLaw DistributionHumans { get; set; }

public CrossRoadLaw DistribustionCars { get; set; }

public uint LightsTime { get; set; }

/// <summary>

/// количество полос кольца

/// </summary>

public int LinesRing { get; set; }

public string Name { get; set; }

public PriorityTypes PriorityType { get; set; }

public int LinesVertical { get; set; }

public int LinesHorisontal { get; set; }

public CrossRoad()

{

}

}

}

/// <summary>

/// клетка среды симуляции

/// </summary>

public class Cell

{

/// <summary>

/// тип клетки по типу (дорога, кольцо)

/// </summary>

public PosTypes TypePosition;

/// <summary>

/// тип клетки по функции

/// </summary>

public FuncTypes TypeFunc;

/\*общее для всех клеток\*/

/// <summary>

/// номер полосы

/// нумерация начинается с единицы по правую сторону от направления движения

/// </summary>

public int LineNumber;

public int X;

public int Y;

/// <summary>

/// машина текущей клетки (null если отсутвует)

/// </summary>

public Car Car;

/// <summary>

/// уровень приоритета клетки: 0 или 1

/// </summary>

public object Priority = 0;

/\*общее для клеток дороги\*/

/// <summary>

/// ссылка на клетку, в которую машина попадет двигаясь вперед по дороге

/// </summary>

public Cell RoadNextCell;

/\*общее для клеток кольца\*/

/// <summary>

/// ссылка на клетку, в которую машина попадет двигаясь вперед по кольцу

/// </summary>

public Cell RingNextCell;

/\*Только для клеток CrossWalk\*/

/// <summary>

/// пешеход, находящийся в клетке пешехдного перехода

/// </summary>

public Human CrosswalkPedestrian;

/// <summary>

/// следующая клетка для перехода по пешеходному переходу

/// </summary>

public Cell CrosswalkNext;

/\*Только для клеток EntryOrDepart\*/

/// <summary>

/// ссылки на дорожки въезда и выезда

/// </summary>

public Cell EntryOrDepartNext;

/// <summary>

/// поизиция выезда или дороги

/// </summary>

public Routes Route;

/// <summary>

/// помещает клетку в позицию X,Y в Environment.CellMap

/// </summary>

public Cell(int X, int Y, PosTypes tp, FuncTypes tf)

{

this.X = X;

this.Y = Y;

TypePosition = tp;

TypeFunc = tf;

Envirmnt.Inst.CellMap[X][Y] = this;

}

public Cell(int X, int Y)

{

this.X = X;

this.Y = Y;

TypePosition = PosTypes.None;

TypeFunc = FuncTypes.None;

}

}

}

public class Car

{

/// <summary>

/// маршрут движения машины

/// </summary>

public Routes RouteTo;

public Routes RouteFrom;

public Cell Location;

public CarStates GoalState;

/// <summary>

/// номер цвета машины

/// </summary>

public int ColorCar;

/// <summary>

/// желаемая линия кольца кольца исходя из RouteFrom и RouteTo

/// </summary>

public int NeedRingLineNumber {

get {

int k = ((int)RouteTo - (int)RouteFrom);

if (k == 0)

return 4;

else if (k < 0)

return 4+k;

else

return k;

}

}

public event EventCarDestroyHandler OnCarDestroy;

public event EventCarMoveHandler OnCarMove;

public Envirmnt Environment;

private static Random \_rand = new Random();

/// <summary>

/// создается новая машина произвольного цвета. Ее состояние - ожидание въезда

/// направление RouteTo задается произвольно

///

/// </summary>

public Car(Cell Location)

{

RouteTo = (Routes)(\_rand.Next(1, 5));

RouteFrom = Location.Route;

this.Location = Location;

GoalState = CarStates.MoveToRing;

//цвета машин

ColorCar = \_rand.Next(0, 4);

}

/// <summary>

/// логика движения машины

/// </summary>

public void TryMoveForward()

{

if (GoalState == CarStates.MoveToRing)

{

if (Location.RoadNextCell != null)

MoveNext(Location.RoadNextCell);

else if (Location.EntryOrDepartNext != null)

MoveNext(Location.EntryOrDepartNext);

if (Location.TypePosition == PosTypes.Ring)

GoalState = CarStates.EntryRing;

}

else if (GoalState == CarStates.EntryRing)

{

if (Location.LineNumber == Envirmnt.Inst.Cross.LinesRing ||

Location.LineNumber >= NeedRingLineNumber)

GoalState = CarStates.MoveToDepart;

else

MoveNext(Location.EntryOrDepartNext);

}

else if(GoalState == CarStates.MoveToDepart)

{

MoveNext(Location.RingNextCell);

if(Location.TypeFunc == FuncTypes.Depart)

{

if (Location.Route == RouteTo)

GoalState = CarStates.DepartRing;

}

}

else if(GoalState == CarStates.DepartRing)

{

MoveNext(Location.EntryOrDepartNext);

if(Location.TypePosition == PosTypes.Road)

GoalState = CarStates.DepartMap;

}

else if(GoalState == CarStates.DepartMap)

{

if (Location.RoadNextCell != null)

MoveNext(Location.RoadNextCell);

else

{

//уничтожение машины

Location.Car = null;

Envirmnt.Inst.Cars.Remove(this);

if (OnCarDestroy != null)

OnCarDestroy(this);

}

}

}

/// <summary>

/// переход на следующую клетку, если там нет пешехода и нет машины

/// </summary>

/// <param name="NextCell">клетка, в которую ехать</param>

public void MoveNext(Cell NextCell)

{

//стоять если пешеход

bool isCrossWalkStop =

(NextCell.TypeFunc == FuncTypes.CrossWalk) &&

((Envirmnt.Inst.Humans.Count > 0) || (Envirmnt.Inst.LightsState == LightStates.Red));

//стоять если машина

bool isCarStop = (NextCell.Car != null);

if (isCrossWalkStop || isCarStop) return;

//изменяем положение машины

Cell CelFrom = Location;

Location.Car = null;

NextCell.Car = this;

Location = NextCell;

//вызываем событие перемещения машины

if (OnCarMove != null)

OnCarMove(this, CelFrom, Location);

}

}

}

public class Envirmnt

{

/// <summary>

/// список всех машин

/// </summary>

public List<Car> Cars;

public List<Human> Humans;

public CrossRoad Cross;

/// <summary>

/// модельное время

/// </summary>

public ulong Time;

/// <summary>

/// матрица клеток окружения

/// первая размерность - X (слева направо)

/// вторая размерность - Y (сверху вниз)

/// нумерация с 0

/// если клетка клетка неактивна (на ней не может быть машин и пешеходов), то элемент = null

/// </summary>

public List<List<Cell>> CellMap;

/// <summary>

/// состояние светофора

/// </summary>

public LightStates LightsState;

/// <summary>

/// объект-одиночка этого типа

/// </summary>

static Envirmnt inst = null;

public static Envirmnt Inst

{

get

{

if (inst == null)

{

inst = new Envirmnt();

}

return inst;

}

}

/// <summary>

/// список создателей новых объектов (машин, пешеходов, сигналов светофора)

/// </summary>

public List<ObjectCreator> Creators;

/// <summary>

/// один шаг основного алгоритма моделирования движения

/// требуется вызывать по таймеру

/// </summary>

public void SimulationStep()

{

//создание новых объектов

foreach (var oc in Creators)

oc.TryCreate();

for (int i = 0; i < Humans.Count; i++)

Humans[i].TryMoveForward();

//учет приоритетов

var ordCars = Cars.OrderBy(car => car.Location.Priority).ToList();

foreach (Car car in ordCars)

car.TryMoveForward();

Time++;

}

}

ПРИЛОЖЕНИЕ Б  
Руководство пользователя

Министерство образования и науки Российской Федерации  
  
Федеральное государственное автономное образовательное учреждение

высшего профессионального образования  
«Самарский государственный аэрокосмический университет

имени академика С. П. Королева

(национальный исследовательский университет)» (СГАУ)

**АВТОМАТИЗИРОВАННАЯ СИСТЕМА МОДЕЛИРОВАНИЯ ДОРОЖНОГО ДВИЖЕНИЯ НА ПЕРЕКРЕСТКЕ С КРУГОВЫМ ДВИЖЕНИЕМ**

**Руководство пользователя**

**ЛИСТ УТВЕРЖДЕНИЯ**

**А.В.00001-01 33 01-1-ЛУ**

Исполнители Мурзин А.В.

Зарецкий А.К.

Руководитель

лабораторной работы Михеева Т.И.

2015

1. Назначение программы

Автоматизированная система моделирования предназначена для имитационного моделирования ситуации на перекрестке с круговым движением, для определения параметров перекрестка, достаточных для обеспечения заданной пропускной способности.

1. Запуск программы

Перед запуском необходимо убедиться, соответствует ли компьютер необходимым требованиям к системе (раздел Б.3).

Программа является переносимым и не требует процедуры установки.

Для установки системы следует запустить файл RoadRingSim.exe либо использовать предварительно созданный ярлык на рабочем столе (рисунок Б.1).



Рисунок Б.1 – Ярлык программы

1. Требования к аппаратным и программным средствам

Минимальные требования к техническим характеристикам и функционированию рабочей станции:

* операционная система: Windows XP и новее;
* процессор – Intel Pentium не менее 1,5 ГГц (от 800 ГГц);
* объем оперативной памяти – 1 Гб (от 450Мб);
* свободное место на диске – 10 Гб (от 2Гб);
* сетевой адаптер – 54/100 Мбит;
* рабочая станция должна иметь доступ в интернет;
* рабочая станция должна быть включена постоянно в течение рабочего времени.

1. Описание основных действий пользователя
   1. Вход в информационную систему

Для начала работы с автоматизированной системой моделирования дорожного движения на перекрестке с круговым движением необходимо запустить приложение двойным щелчком по ярлыку рабочего стола RoadRingSim или по файлу RoadRingSim.exe.

После запуска системы откроется окно, предназначенное для авторизации пользователей (рисунок Б.2). Логины и пароли устанавливаются суперпользователем.

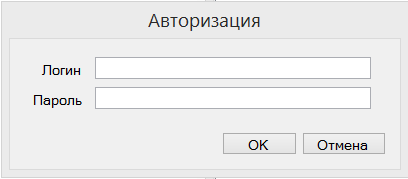


Рисунок Б.2 – Форма авторизации пользователей

В случае неверно указанных данных, система выведет сообщение об ошибке, форма которого представлена на рисунке Б.3, где, нажав кнопку «ОК», можно вернуться на форму авторизации и повторно ввести имя и пароль пользователя.

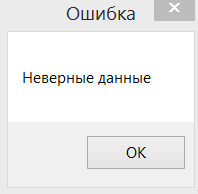


Рисунок Б.3 – Сообщение об ошибке авторизации

* 1. Работа с моделями перекрестков

После успешного прохождения авторизации, открывается окно выбора модели перекрестка, моделирование движения на котором будет осуществляться (рисунок Б.4).

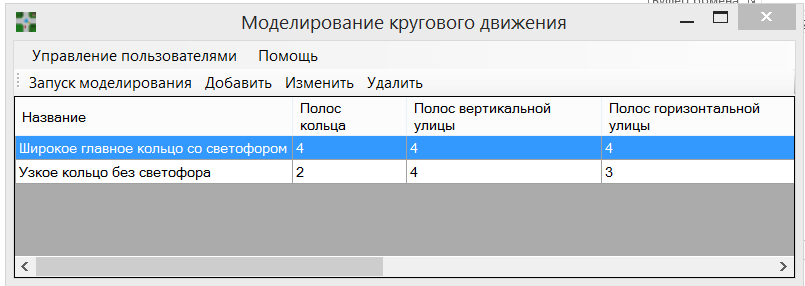


Рисунок Б.4 – Окно выбора модели перекрестка

Для запуска моделирования необходимо выделить в списке нужную модели и нажать пункт меню «Запуск моделирования». В этом окне администратор имеет возможность управлять моделями (удалять, изменять, добавлять), а также перейти к управлению пользователями системы.

* 1. Процесс моделирования

После запуска моделирования откроется окно процесса моделирования. Большую часть окна занимает визуализация процесса моделирования. Прямоугольники разных цветов являются условным обозначением автомобилей, квадраты разных цветов – обозначением пешеходов. В нижней части находится условное изображение сигнала светофора для автомобилей. Вдоль дороги располагаются дорожные знаки, смысл которых регламентируется в ПДД.

Слева находится панель управления моделированием: ползунок управления скоростью, кнопки запуска и приостановки моделирования.

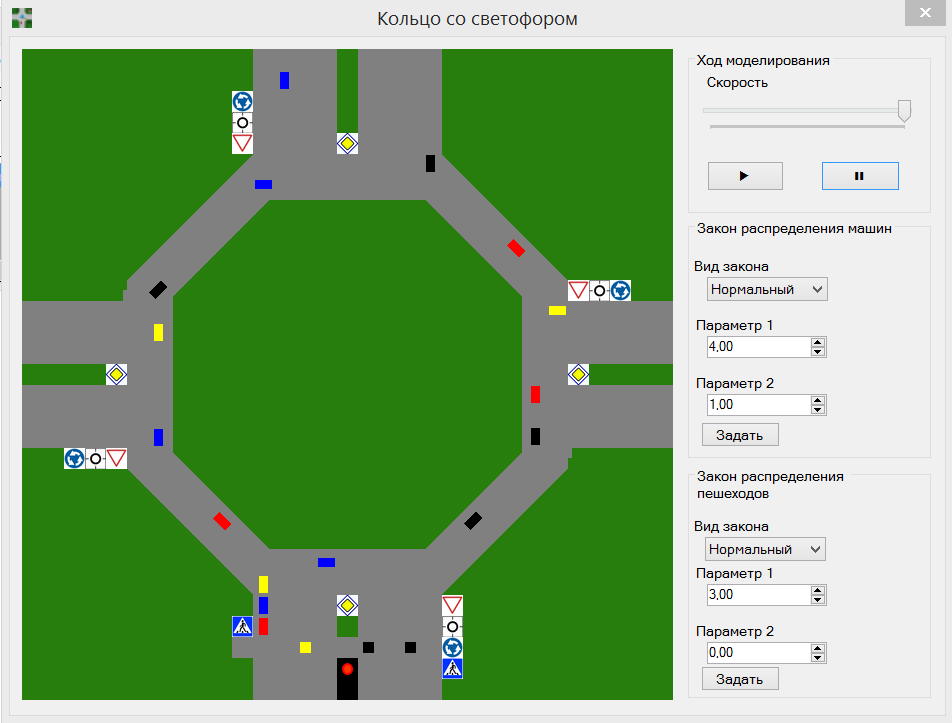


Рисунок Б.5 – Окно моделирования

Задание законов распределения автомобилей и пешеходов доступно для расширенных пользователей и администраторов (см. Б.4.4). После изменения вида закона и/или параметра необходимо зафиксировать эти изменения кнопкой «Задать».

При выборе нормального закона распределения Параметр 1 используется в качестве математического ожидания (среднего значения), а Параметр 2 – в качестве дисперсии. При выборе экспоненциального закона распределения учитывается только Параметр 1 и он играет роль интенсивности событий появления автомобилей/пешеходов. При выборе равномерного закона распределения интервалы времени между появлением новых автомобилей/пешеходов будут являться любым целым значением из полуинтервала [параметр 1, параметр 2) с равной вероятностью.

Законы распределения определяют интервалы модельного модельного времени, а физическое время зависит от положения ползунка скорости моделирования и производительности ЭВМ. Значения, заданные в этом окне, не фиксируются в базе данных, а действуют только на время текущего сеанса работы.

* 1. Разграничение прав доступа

В системе организовано разграничение прав доступа. При входе в АИС необходимо ввести в диалоговом окне логин и пароль. В системе имеется три пользовательские роли.

Пользователь имеет возможность только запускать моделирования без возможности измения каких-либо параметров.

Расширенный пользователь имеет дополнительную возможность изменять распределение автомобилей и пешеходов прямо во время моделирования.

Суперпользователь имеет полный набор прав доступа. Помимо возможностей расширенного пользователей, имеет доступ к управлению пользователями и моделями перекрестков: созданию, изменению, удалению.

1. Описание основных действий суперпользователя
   1. Управления моделями перекрестков

Для суперпользователя в окне выбора перекрестка доступны функции удаления, перекрестков, а также вызова окна создания и изменения модели перекрестка.

В окне изменения модели перекрестка, Рисунок Б.6, доступно редактирование всех параметров модели. После нажатия кнопки изменить изменения зафиксируется в базе данных. Окно создания модели полностью аналогично.

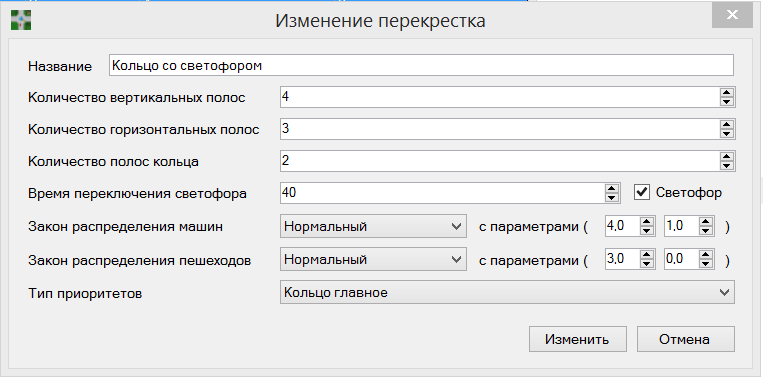


Рисунок Б.6 – Окно изменения модели перекрестка

Если один из параметров при создании или изменении модели не будет задан, то появится сообщение об ошибке и незаполненные поля будут обозначены красным восклицательным знаком, рисунок Б.7.

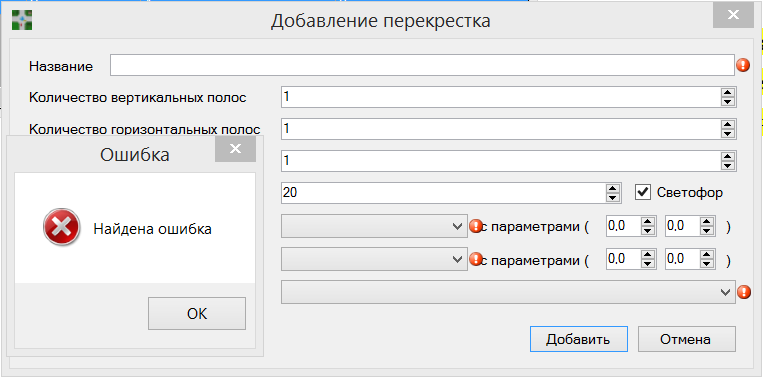


Рисунок Б.7 – Ошибка при работе с моделями перекрестков

* 1. Управления пользователями

Суперпользователь имеет возможность управлять зарегистрированными в системе пользователями. Для этого в окне выбора модели перекрестка необходимо выбрать пункт меню «Управление пользователями».

Откроется окно менеджера пользователей, рисунок Б.7. В этом окне отображаются список существующих пользователей.

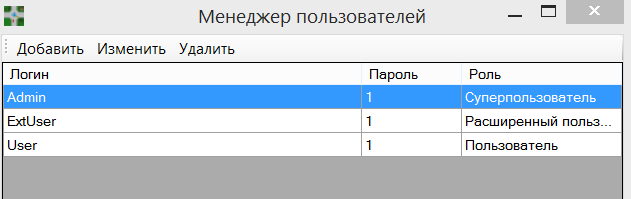


Рисунок Б.8 – Окно менеджера пользователей

Пункт меню изменить (добавить) открывает окно редактора пользователя, рисунок Б.9.

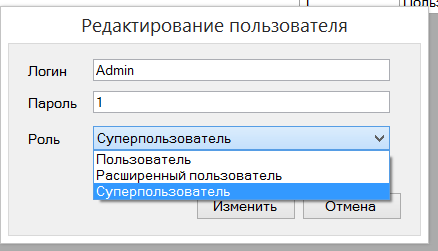


Рисунок Б.9 – Окно редактора пользователя.

В системе предусмотрена защита от удаления последнего администратора. Не допускается создавать пользователей с пустым логином и/или паролем и нескольких пользователей с одинаковым логином.