Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Кафедра «Информатика»
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой

_____ _____
подпись инициалы, фамилия
«_____» _____20__ г.

БАКАЛАВРСКАЯ РАБОТА

09.03.04 «Программная инженерия»
код и наименование специальности
Программная модель имитации процесса регулирования транспортного
тема
движения с применением взаимодействующей адаптивной системы управления работы светофоров

Руководитель _____ _____ _____
подпись, дата должность, ученая степень инициалы, фамилия
Выпускник _____ _____
подпись, дата инициалы, фамилия
Консультант _____ _____ _____
подпись, дата должность, ученая степень инициалы, фамилия
Нормоконтролер _____ _____
подпись, дата инициалы, фамилия

Красноярск 2019

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий
институт
Кафедра «Информатика»
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой

_____ _____
подпись инициалы, фамилия
«_____» _____20__г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Студенту Радионову Тимофею Вадимовичу

фамилия, имя, отчество

Группа КИ15-16Б Направление (специальность) 09.03.04

номер код

Программная инженерия

наименование

Тема выпускной квалификационной работы: «Программная модель имитации процесса регулирования транспортного движения с применением взаимодействующей адаптивной системы управления работы светофоров»

Утверждена приказом по университету №_____ от _____

Руководитель ВКР: А.С. Михалев, старший преподаватель, ИКИТ СФУ

Исходные данные для ВКР: справочная, научная и методическая литература, ресурсы Интернет

Перечень разделов ВКР: введение, анализ предметной области, спецификация требований к программному продукту, характеристика обеспечивающих элементов объекта проектирования, процессы проектирования программного обеспечения, описание разработанного программного продукта, исследование алгоритмов регулирования светофоров, заключение

Перечень графического материала ВКР: презентационные слайды PowerPoint

Руководитель ВКР _____ А.С. Михалев

подпись инициалы, фамилия

Задание принял к исполнению _____ Т.В. Радионов

подпись инициалы, фамилия

«___» _____ 20__ г.

РЕФЕРАТ

Выпускная квалификационная работа по теме «Программная модель имитации процесса регулирования транспортного движения с применением взаимодействующей адаптивной системы управления работы светофоров» содержит 78 страниц текстового документа, 23 использованных источника, { иллюстрация, 8 таблиц.

Целями данной работы являются:

- разработка программной модели дорожного движения для имитационного моделирования работы алгоритмов регулирования светофоров;
- программная реализация алгоритмов регулирования дорожного движения (ССУРС, АСУРС и ВАСУРС с целью апробации программной модели);
- исследование и сравнение регулирования дорожного движения по эффективности и временным затратам.

В первом разделе производится анализ предметной области и делается заключение о целесообразности разработки программного продукта. Во втором разделе предъявляются спецификации требований к программному продукту. В третьем разделе приводится характеристика обеспечивающих элементов объекта проектирования: выбираются инструменты разработки и способ управления проектом. В четвертом разделе описываются процессы проектирования программного продукта: этапы проектирования, архитектура, алгоритмы, тестирование, интерфейсы, база данных. В пятом разделе описывается разработанный программный продукт. В шестом разделе производится исследование разработанных алгоритмов.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Каждый город, испытывающий затруднения с транспортным движением, нуждается в инновационных способах для разрешения ситуации на дорогах. Количество автомобилей с каждым днем только возрастает, а также растет и количество проблем на дорогах. Причины заторов на дорогах разнообразны: начиная от проблем самих дорог и заканчивая рядовыми водителями транспортных средств [1]. К одной из основных проблем относится неправильная организация дорожного движения. В особенности, к этому относится регулирование движения с помощью светофоров [2].

Развитие транспортной инфраструктуры крупных городов требует создания интегрированных систем управления нового поколения, позволяющих определять оптимальные режимы движения транспорта с учетом изменчивости дорожной обстановки. Усовершенствование работы светофоров может оказать значительное влияние на дорожную ситуацию в городе в целом [3].

В настоящее время большинство светофоров на территории Российской Федерации работают по заранее настроенным программам смены сигнала [4] (далее в работе такой алгоритм регулирования сигнала будут упоминаться как ССУРС). Данный тип светофоров работает как простой механизм смены сигналов с настроенным таймером под каждый из них. Существуют также модификации данных светофоров, однако суть их неизменна: они недостаточно гибки по отношению к дорожной ситуации и сигналы регулируются согласно ССУРС [5].

Давно не в новинку уже адаптивные светофоры, которые подстраивают переключение сигналов под регулируемый ими поток машин с помощью дополнительных устройств (далее в работе такой алгоритм регулирования сигнала будут упоминаться как АСУРС). Данный тип светофоров до сих пор мало распространен в Российской Федерации, однако это вопрос времени [6].

При решении проблемы организации городского движения и управления транспортными потоками широко используют элементы интеллектуальной

транспортной инфраструктуры. В основе работы любой интеллектуальной системы управления дорожным движением лежит алгоритм оптимизации управления сигналами светофоров [7].

Каждый новый разработанный алгоритм регулирования сигналов светофоров нуждается в предварительной проверке и настройке до введения в эксплуатацию. Обычно это делают путем моделирования транспортного движения, где также присутствуют моделируемые объекты-светофоры, которыми, непосредственно, и должны управлять данные алгоритмы [8]. Однако разработчикам зачастую приходится составлять собственную имитационную модель транспортного движения и регулировки сигналов светофора, поскольку существующие решения не могут полностью обеспечить работу алгоритма [9].

В рамках данной ВКР была сделана попытка разработать собственный алгоритм регулирования светофоров (далее в работе такой алгоритм регулирования сигнала будут упоминаться как ВАСУРС).

Цели данной работы:

- разработка программной модели дорожного движения для имитационного моделирования работы алгоритмов регулирования светофоров;
- программная реализация алгоритмов регулирования дорожного движения (ССУРС, АСУРС и ВАСУРС с целью апробации программной модели);
- исследование и сравнение регулирования дорожного движения по эффективности и временным затратам.

Для достижения поставленной цели необходимо:

- проанализировать предметную область;
- определить спецификации требований к программному продукту;
- определить обеспечивающие элементы объекта проектирования;
- выполнить процессы проектирования программного обеспечения;
- описать разработанный программный продукт;

- выполнить исследование алгоритмов регулирования светофоров на разработанном программном продукте;

- сделать выводы по результатам исследования.

Полученный в ходе разработки программный продукт может быть использован для разработки и исследования различных алгоритмов регулирования светофоров, а описанный алгоритм ВАСУРС, если по результатам исследований покажет свою значимость, может быть использован на реальных системах регулирования светофоров.

Проведем анализ предметной области в следующем разделе.

1 Анализ предметной области

В данном разделе осуществлено исследование и анализ предметной области с целью определить основы проекта, рассмотреть существующие аналоги и сделать вывод о необходимости и о возможности реализации решения.

1.1 Описание предметной области

За основу работы взята проблема разработки и исследования инновационных алгоритмов регулирования светофоров, являющихся частью ИТС, посредством моделирования с целью дальнейшего улучшения и реального применения в существующих системах управления светофорами.

Алгоритм регулирования светофора – управляющая программа, обеспечивающая последовательное переключение состояний сигнала согласно внутреннему коду [10].

Под инновационными алгоритмами регулирования светофорами подразумеваются такие алгоритмы, которые еще не применяются на практике, однако обладают качествами, которые делают работу светофоров более эффективной, чем у светофоров с ныне существующими алгоритмами.

Под ИТС подразумевается система управления, реализующая инновационные разработки для управления автомобильными потоками.

Эффективность работы светофора можно оценивать посредством того, насколько они допускают возникновение транспортных пробок – одного из наиболее негативных факторов современного города. Их отрицательное влияние распространяется на множество аспектов, таких как логистика, производительность труда, экология и многие другие [6].

Под моделированием рассматривается процесс имитации какой-либо ситуации или процесса, с целью получения объяснений этих явлений, а также для предсказания явлений, интересующих исследователя. В данном случае будет моделироваться движение транспортного потока и его регулировка смоделированными светофорами.

Выделяют три вида моделирования дорожного движения [11]:

а) макромоделирование – тип моделирования, основывающийся на применении к автомобильному трафику законов гидродинамики, по аналогии с жидкостью в трубе. Как следствие, такой тип моделирования выражается в написании систем дифференциальных уравнений в частных производных, сформулированных относительно интересующих величин – например, плотности потока автомобилей или их средней скорости;

б) микромоделирование – каждый автомобиль представляется индивидуально, и потому для каждого автомобиля решается уравнение (обычно простое дифференциальное). Преимущество микромоделирования состоит в возможности представления перегруженных дорожных сетей, поскольку микромоделирование позволяет симулировать очереди;

в) мезомоделирование – способ моделирования, в котором предлагается определить функцию $f(t, x, V)$, которая выражает вероятность нахождения автомобиля, движущегося со скоростью V, во время t в точке x. Далее эта функция может быть вычислена с использованием методов статистической механики – решением интегро-дифференциального уравнения, например, уравнения Больцмана.

Для анализа дорожной обстановки в городе при плотном потоке движения с применением светофоров предпочтительнее всего микромоделирование [11].

Рассмотрим существующие решения, которые предназначены для моделирования дорожного движения и в них присутствует объект-модель светофор.

1.2 Обзор существующих решений

Одним из популярных решений для моделирования дорожного движения является программа AnyLogic [12], а именно ее библиотека дорожного движения, которая позволяет детально планировать, проектировать и моделировать транспортные потоки с учетом индивидуального поведения каждого водителя.

Алгоритмы движения в AnyLogic настроены в соответствии с правилами дорожного движения — учет ограничения скорости, «уступи дорогу» и так далее. В то же время, в моделях дорожного движения каждое транспортное средство представляется в виде агента, который имеет индивидуальные физические параметры и поведенческие шаблоны. Кроме того, у пользователя есть возможность создавать в модели двухмерную и трехмерную анимацию транспортных средств и окружающих объектов.

Библиотека дорожного движения в AnyLogic – это инструмент планирования и организации транспортных потоков. В моделях дорожного движения имитируется перемещение машин по улицам и автомагистралям, включая такие элементы как перекрестки, пешеходные переходы, круговое движение, автостоянки и остановки общественного транспорта. Возможности библиотеки позволят решить следующие задачи:

- планирование дорог и автомагистралей;
- оценка загруженности и пропускной способности дорог;
- оптимизация фаз светофоров;
- интеграция общественных зданий в дорожную сеть.

Однако данное решение ограничено встроенным инструментарием для регулирования светофоров (рисунок 1) и является платным продуктом. Даже дополнительная библиотека моделирования процессов не позволяет, к примеру, организовать взаимодействие между светофорами, которое требуется для работы алгоритма ВАСУРС.

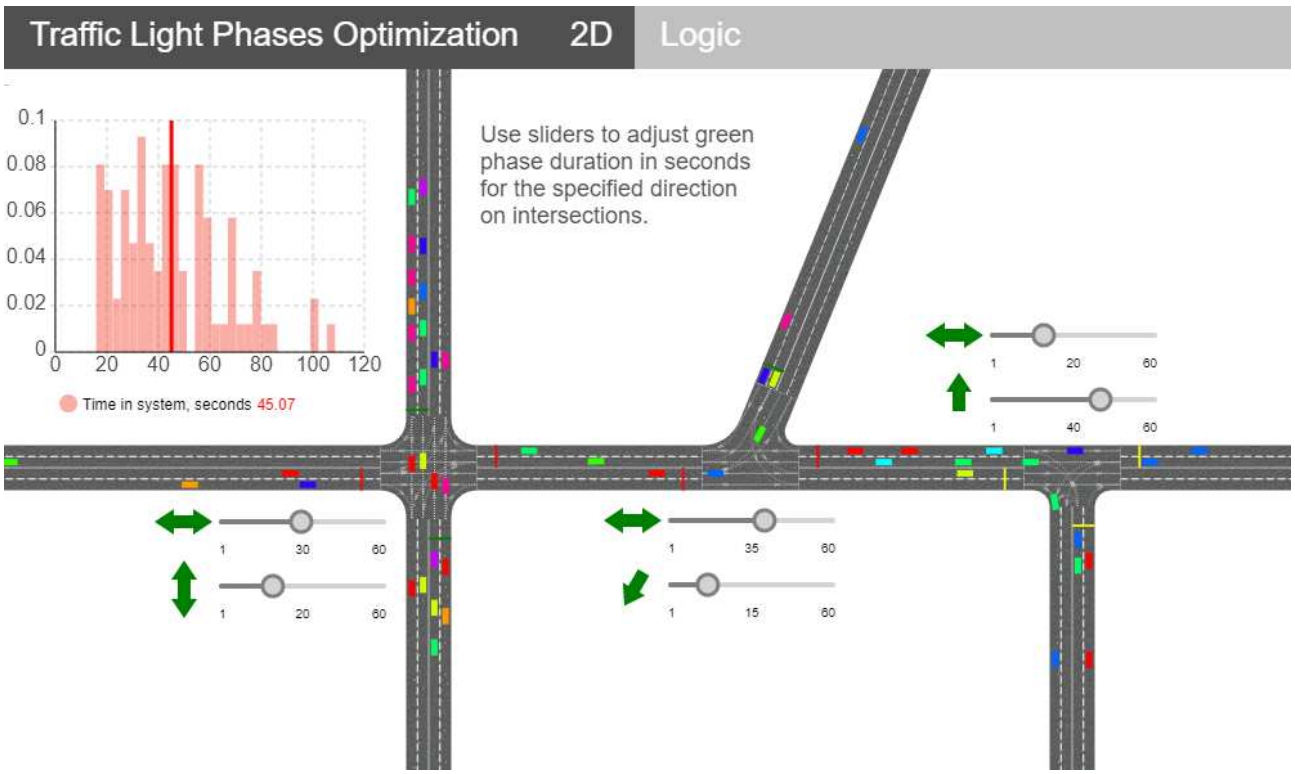

Рисунок 1 – Пример управления светофорами в AnyLogic

PTV Vistro [13] является наиболее подходящим инструментом для моделирования регулирования светофорами дорожного движения.

Основными достоинствами программного продукта PTV Vistro, выгодно отличающими его от других программных продуктов, является возможность оптимизации режимоврегулирования, включая одновременную оптимизацию и координацию сразу нескольких регулируемых пересечений (кольцевых, нерегулируемых, регулируемых).

Однако данный продукт является платным, хотя и есть возможность получить студенческую лицензию, но в таком случае на программу наложены ограничения. В PTV Visio при работе с светофорами можно выполнять лишь оптимизацию их сигналов (рисунок 2), но не применять собственные алгоритмы.

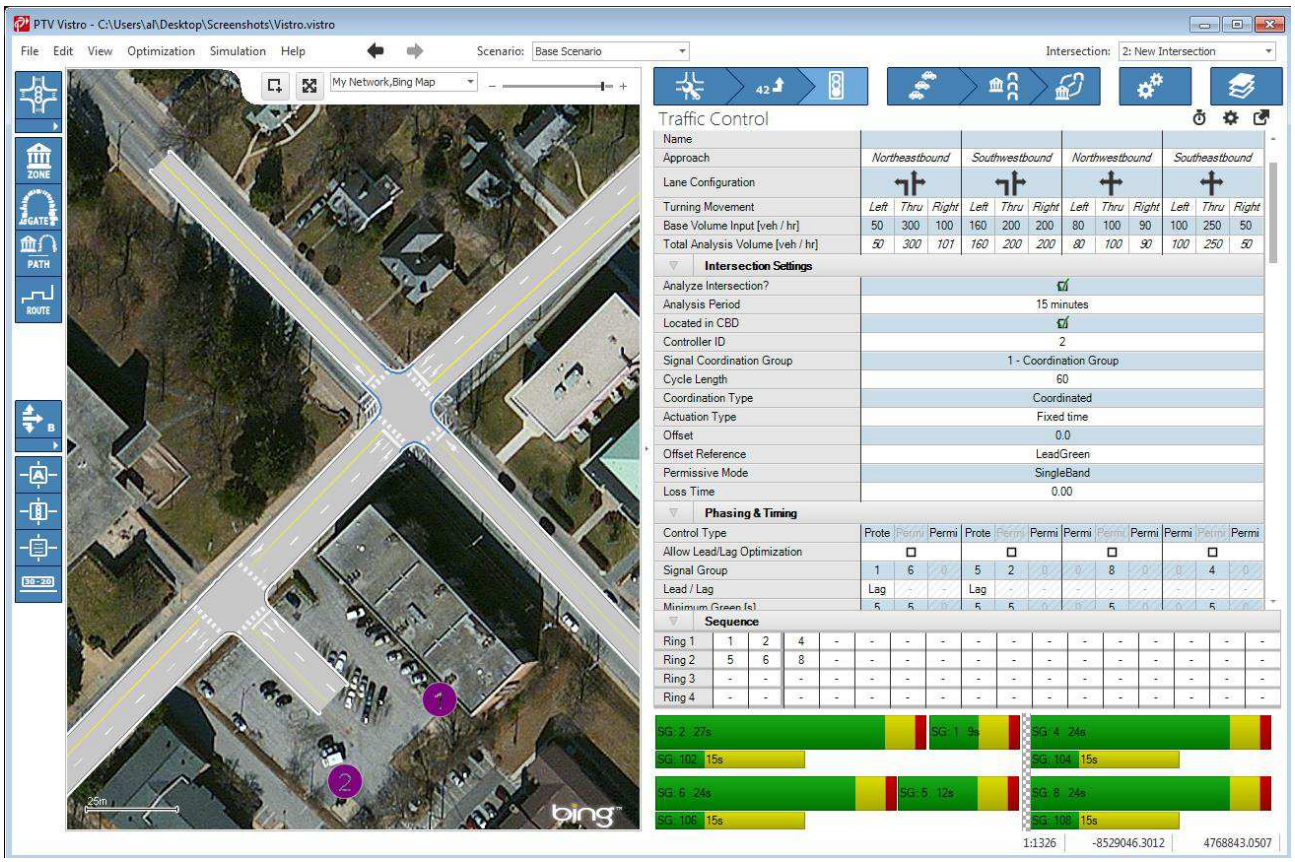

Рисунок 2 – Пример управления светофорами в PTV Vistro

Aimsun [14] (расширенный интегрированный микроскопический симулятор для городских и внегородских сетей) – программа моделирования дорожного движения разработана и предназначена для анализа дорожного движения при проектировании новых и оценки существующих проектных решений. Aimsun позволяет моделировать адаптивные системы управления дорожным движением (рисунок 3), системы приоритетного движения общественного транспорта, позволяет выполнять экологическую оценку воздействия транспорта на окружающую среду. Колоссальный инструмент для моделирования (микромоделирование, мезомоделирование, макромоделирование) с очень большим функционалом.

Из недостатков можно выделить, что данный продукт, как и предыдущие, является платным, в нем отсутствует возможность управлять светофорами собственными алгоритмами, а также данный продукт является зарубежным решением и, как следствие, не имеет русскоязычного интерфейса.

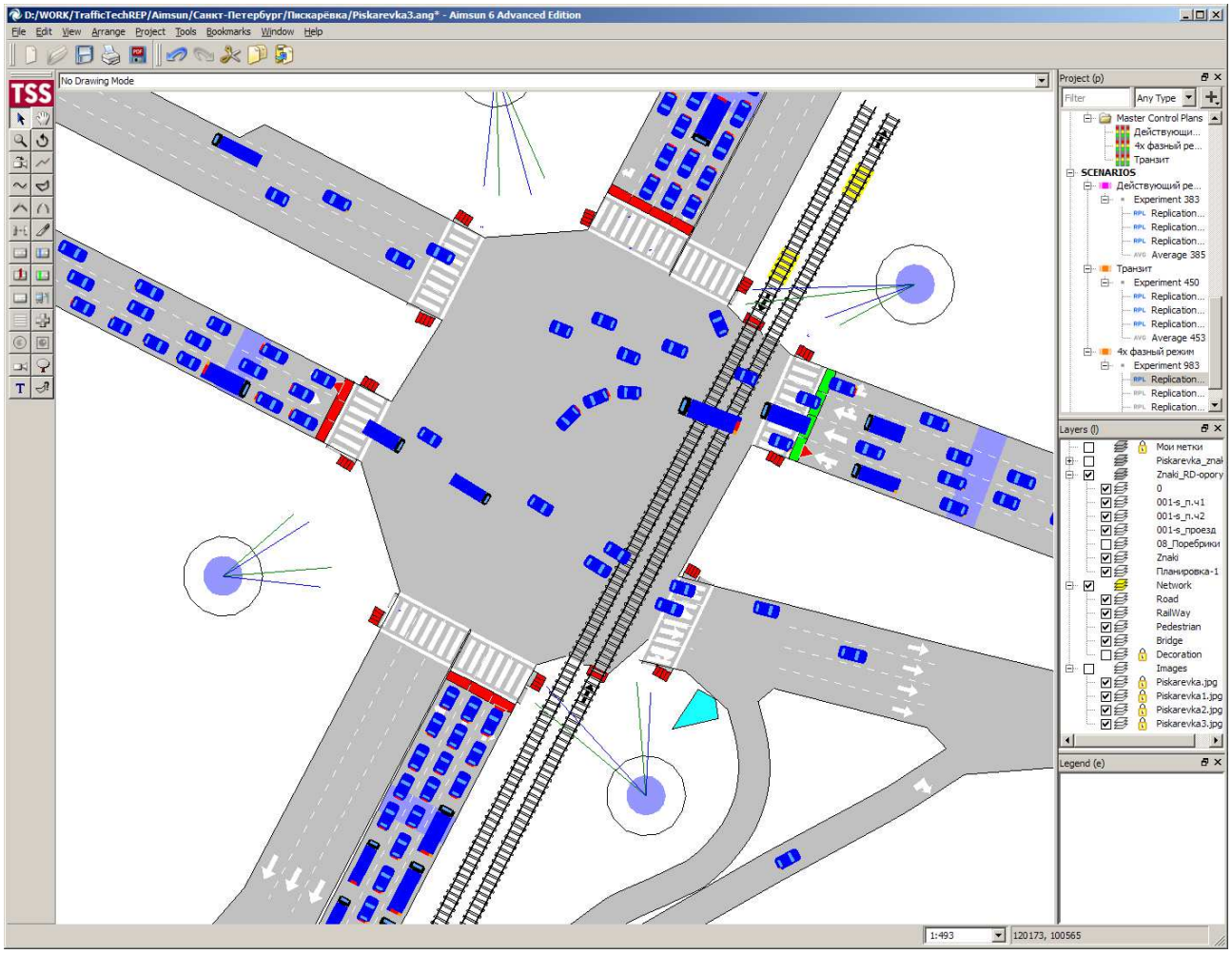

Рисунок 3 – Пример управления светофорами в Aimsun

Подведем итоги и сделаем заключение о целесообразности разработки собственного продукта.

1.3 Заключение

Исходя из вышеописанных существующих решений можно сделать вывод, что большинство программных продуктов, специализирующихся на моделировании движения трафика и управлением светофорами, являются платными решениями, для обычного пользователя таких программ отсутствует доступ к исходному коду и, как следствие, пользователь ограничен возможностями лишь того инструментария, что присутствует в программе.

В случае задачи разработки и исследования новых алгоритмов регулирования светофоров, то в аналогах большинство инструментов и библиотек не будет использоваться. А те библиотеки, которые действительно

потребуются, отсутствуют. По причине того, что нет доступа к исходному коду, у пользователя нет возможности написать свою библиотеку для существующего программного продукта с целью его модификации под свои нужды.

В результате исследования предметной области и обзора существующих решений можно сделать заключение, что для разработки нового алгоритма регулирования светофоров, такого как ВАСУРС, необходимо самостоятельно разработать собственную программную модель имитации дорожного движения на регулируемом перекрестке, удовлетворяющей поставленной цели. Такая программная модель должна обеспечить возможность подключать и иные алгоритмы регулирования светофоров. Также стоит рассмотреть вопрос об открытом исходном коде программы с целью обеспечить возможность другим пользователям выполнять задачи схожего характера.

2 Спецификация требований к программному продукту

Далее представлена спецификация требований согласно стандарту IEEE 830 (структура SRS) [15].

2.1 Введение

Цели:

Получить программную модель движения автомобилей на перекрестках для демонстрации эффективности различных алгоритмов регулирования светофоров при одинаковых условиях с целью выявить наилучший из примененных алгоритмов.

Соглашение о терминах:

Полное наименование программного продукта: «Имитационная модель процесса регулирования транспортного движения»

Условное обозначение: ИМПРТД

Предполагаемая аудитория и последовательность восприятия:

Предлагаемая аудитория и последовательность восприятия отображена на матрице влияния (таблица 1).

Таблица 1 – Матрица влияния

Группа «A»		Группа «B»		Группа «C»		Группа «D»	
Разработчик проекта	5	Дорожные службы города	3	Разработчики алгоритмов светофоров	4	Пешеходы	2
Компании, занимающиеся имитационным моделированием регулируемого движения	3	Консультанты	4	Разработчики прочего ПО	3	Водители	3
1 – Неосведомленность; 2 – Сопротивление; 3 – Нейтралитет; 4 – Поддержка; 5 – Активное участие							

Стратегия работы с заинтересованными сторонами основана на матрице «Интерес-Влияние» (рисунок 4).

Рисунок 4 – Матрица «Интерес-Влияние»

Группа «A» - Высокое влияние и высокий интерес

Стратегия заключается в максимальном вовлечении. Данная группа представляет собой основных стейкхолдеров проекта и должна максимально привлекаться к принятию решений в проекте. Необходимо повышать заинтересованность группы в проекте и полностью удовлетворять ее потребности. Рекомендуется использовать принцип партнерства в коммуникации при ведении переговоров по проекту с этой группой.

Группа «B» - Высокое влияние и низкий интерес

Стратегия, которая носит консультативный характер. Их рекомендуется привлекать в качестве консультантов и согласовывать с ними только важные стратегические решения по проекту.

Группа «C» - Низкое влияние и высокий интерес

Стратегия заключается в получении поддержки проекта. Данная группа стейкхолдеров должна быть ознакомлена со всеми ключевыми решениями по проекту несмотря на то, что она не принимает прямого участия в решениях по проекту. При этом рекомендуется данную группу привлекать к обсуждению возможных проблем и заручаться поддержкой у нее дополнительной поддержкой по важным решениям.

Группа «D» - Низкое влияние и низкий интерес

Стратегия заключается в игнорировании. Рекомендуется исключительно привлекать данную группу к выполнению требуемых задач, не погружать ее в детали проекта и использовать самый низкий уровень информирования.

Масштаб проекта:

Определены следующие границы проекта:

- данные об автомобилях, светофорах хранятся в базе данных, причем данные об автомобиле необходимы до тех пор, пока он не покинет область модели;

- светофор хранит данные об автомобилях до тех пор, пока они не выйдут из его области действия;

- автомобиль знает данные лишь о встречном автомобиле, который пересекает траекторию его движения перед ним и/или знает данные об движущемся перед ним автомобиле;

- задача каждого автомобиля достигнуть конечной своей точки по правилам модели, светофора и взаимодействия с другим автомобилем.

Ссылки на источники:

Представлен в разделе «список использованной литературы».

2.2 Общее описание

Видение продукта:

Программный продукт должен представлять из себя приложение, в котором можно будет имитировать движение транспорта и работу светофоров. Весь процесс имитации должен быть управляемым и наглядным. Также в данном продукте необходима возможность редактирования необходимых элементов и возможность выбора алгоритма.

Функциональность продукта:

- управление настройками модели;

- отображение окна с визуализацией моделируемого процесса;

- настройка маршрутов для движения транспорта;

- настройка параметров светофора;
- переключать алгоритмы регулирования светофоров;
- имитация движения транспорта и работы светофоров.

Также для наглядности составлена Use-Case диаграмма (рисунок 5).

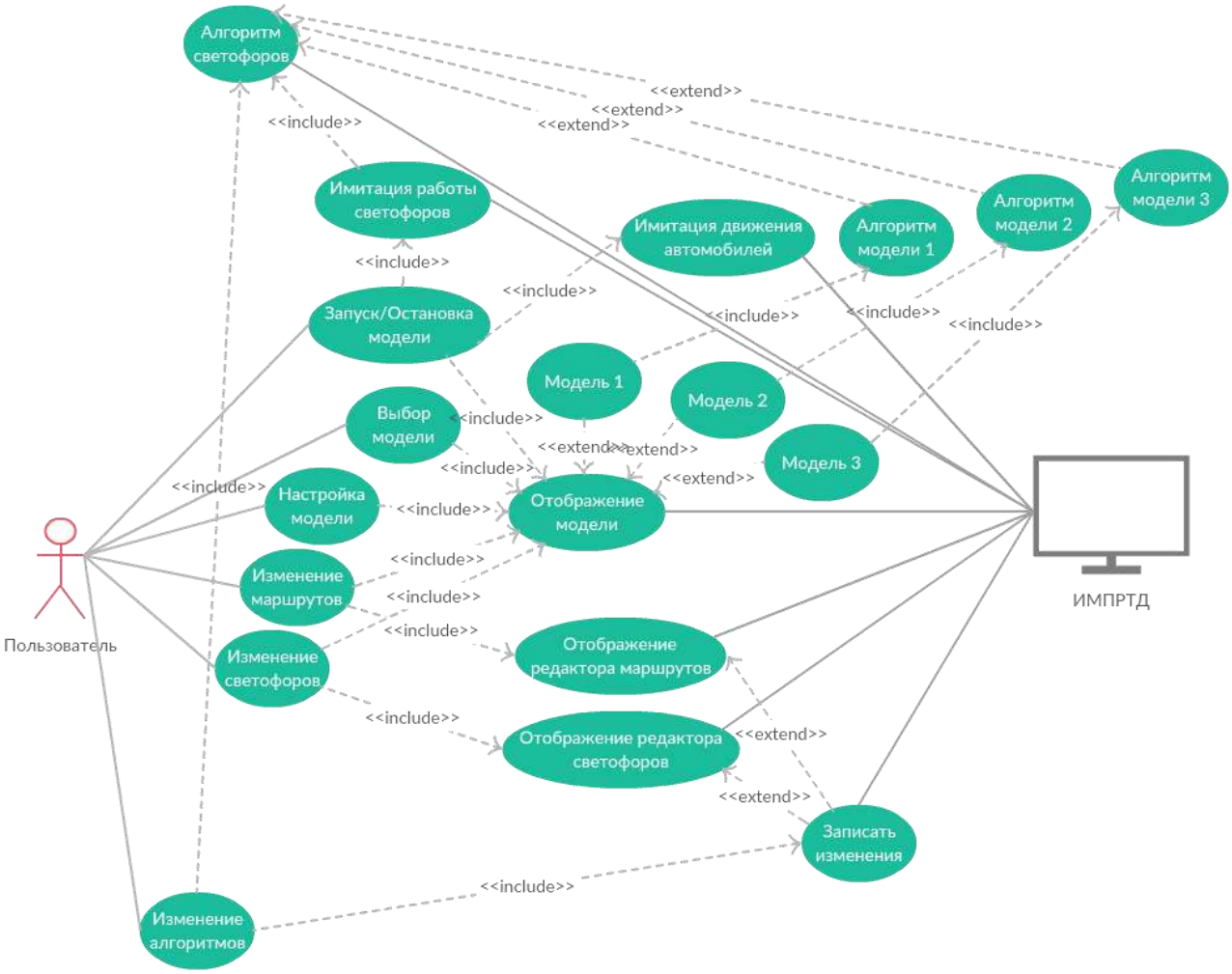

Рисунок 5 – Use-Case диаграмма программного продукта

Классы и характеристики пользователей:

Для использования ИМПРТД отсутствует разделение на классы пользователей.

Пользователь должен иметь представление о том, как передвигаются автомобили и какими правилами они руководствуются при работе с моделью, то есть обладать знаниями о процессе движения на регулируемом перекрестке.

Также пользователь должен обладать знаниями в программировании и ему необходимо наличие опыта работы с языком программирования, на котором будет составлен программный продукт.

Среда функционирования продукта:

В качестве среды функционирования выбрано устройство под управлением наиболее распространенной ОС: Windows, начиная от версии XP и выше.

Рамки, ограничения, правила и стандарты:

Продукт не должен копировать частично или полностью существующие решения, то есть не нарушать авторских прав.

Продукт должен использоваться на устройстве, которое на аппаратном уровне совместимо с ОС, для которой продукт предназначен.

Внутренний код продукта должен соответствовать общепринятым стандартам языка программирования, на котором он будет написан.

Продукт должен иметь защиту от ввода неверных данных или от действий, которые могут привести к неправильной работе программы.

Исходный код продукта не должен быть в открытом доступе до тех пор, пока не выйдет на стадию «Релиз».

Документация для пользователей:

Документация для пользователя будет представлена в виде руководства пользователя.

Допущения и зависимости:

Допускается, что программа будет использовать оперативную память для хранения временных данных и место на жестком диске для хранимой информации.

Предполагается, что производительность программного продукта будет зависеть от производительности процессора.

2.3 Функциональность продукта

Функциональный блок управления настройками модели:

Описание и приоритет: данный блок отвечает за основные настройки работы программы, которые задаются непосредственно пользователем. Является активным при запуске программного продукта.

Причинно-следственные следственные связи, алгоритмы: данный блок запускается при старте программы, из него запускаются остальные блоки. В нем описаны действия при использовании UI-элементов данного блока. При завершении работы с блоком завершается программа.

Функциональные требования: контроль ввода данных пользователя и оповещения, считывание настроек или иных данных при запуске, хранение настраиваемых параметров до завершения приложения, контроль остальных запускаемых окон приложения, управление UI-объектами данного блока.

Функциональный блок отображения окна с визуализацией моделируемого процесса:

Описание и приоритет: данный блок отвечает за вывод и визуализацию результата работы запускаемых алгоритмов. Приоритет запуска задается блоком управления моделью.

Причинно-следственные следственные связи, алгоритмы: данный блок запускается при старте программы, а также перезапускается при выборе в блоке управления какой-либо модели или при запуске блоков настраивания маршрутов или светофоров. В нем описаны алгоритмы прорисовки объектов, которые необходимо отобразить, а также описан алгоритм масштабирования управляемого им окна.

Функциональные требования: обновление состояния с заданным интервалом, контроль данных перед отображением.

Функциональный блок настройки маршрутов для движения транспорта:

Описание и приоритет: данный блок отвечает за отображение и изменение данных, связанных с объектами модели – маршрутами. Приоритет запуска задается блоком управления моделью.

Причинно-следственные следственные связи, алгоритмы: данный блок запускается при соответствующем действии пользователя в блоке управления моделью. В нем описаны действия при использовании UI-элементов данного блока. При завершении работы с блоком управление программы передается блоку управления моделью.

Функциональные требования: считывание, изменение, сохранение и отображение данных о маршрутах. Контроль ввода данных и обновление состояния блока отображения окна визуализации моделируемого процесса.

Функциональный блок настройки параметров светофора:

Описание и приоритет: данный блок отвечает за отображение и изменение данных, связанных с объектами модели – светофорами. Приоритет запуска задается блоком управления моделью.

Причинно-следственные следственные связи, алгоритмы: данный блок запускается при соответствующем действии пользователя в блоке управления моделью. В нем описаны действия при использовании UI-элементов данного блока. При завершении работы с блоком управление программы передается блоку управления моделью.

Функциональные требования: считывание, изменение, сохранение и отображение данных о светофорах. Контроль ввода данных и обновление состояния блока отображения окна визуализации моделируемого процесса.

Функциональный блок переключения алгоритмов регулирования светофоров:

Описание и приоритет: данный блок является внутренним блоком управления моделью и отвечает за переключение моделей и изменение соответствующих алгоритмов работы светофора. Приоритет запуска задается блоком управления моделью.

Причинно-следственные следственные связи, алгоритмы: данный блок выполняется при соответствующем действии пользователя в блоке управления моделью. При его запуске происходит повторное считывание данных о светофорах для соответствующей модели и использование их алгоритмов.

Функциональные требования: считывание данных о светофорах определенной модели. Обновление состояния блока отображения окна визуализации моделируемого процесса.

Функциональный блок имитации движения транспорта и работы светофоров:

Описание и приоритет: данный блок является внутренним блоком управления моделью и отвечает за выполнение алгоритмов движения транспорта и работы светофоров на модели с заданным интервалом времени. Приоритет запуска задается блоком управления моделью.

Причинно-следственные следственные связи, алгоритмы: данный блок выполняется при соответствующем действии пользователя в блоке управления моделью. При его запуске происходит периодический запуск алгоритмов движения транспортов и светофоров. Также при соответствующем действии пользователя в блоке управления моделью происходит остановка вышеперечисленных алгоритмов.

Функциональные требования: вычисление значений функций связанных алгоритмов, старт и остановка выполнения связанных алгоритмов. Обновление состояния блока отображения окна визуализации моделируемого процесса.

2.4 Требования к внешним интерфейсам

Интерфейсы пользователя:

Окно управления настройками модели (рисунок 6).

Здесь необходимы панель переключения моделей, кнопки запуска и остановки выполнения моделирования. Также нужны UI-элементы для ввода необходимых данных. Дополнительно добавить фильтры для отображения

необязательных, но полезных элементов на модели. Необходимы кнопки для отображения окон настроек маршрутов и светофоров.

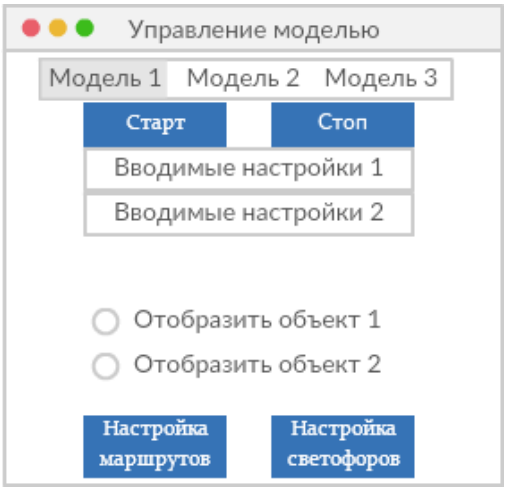

Рисунок 6 – Окно управления настройками модели

Окно визуализации моделируемого процесса (рисунок 7)

Данное окно должно быть достаточных размеров для восприятия и подстраиваться под экран устройства. Оно не взаимодействует с пользователем и является лишь местом визуализации моделирования.

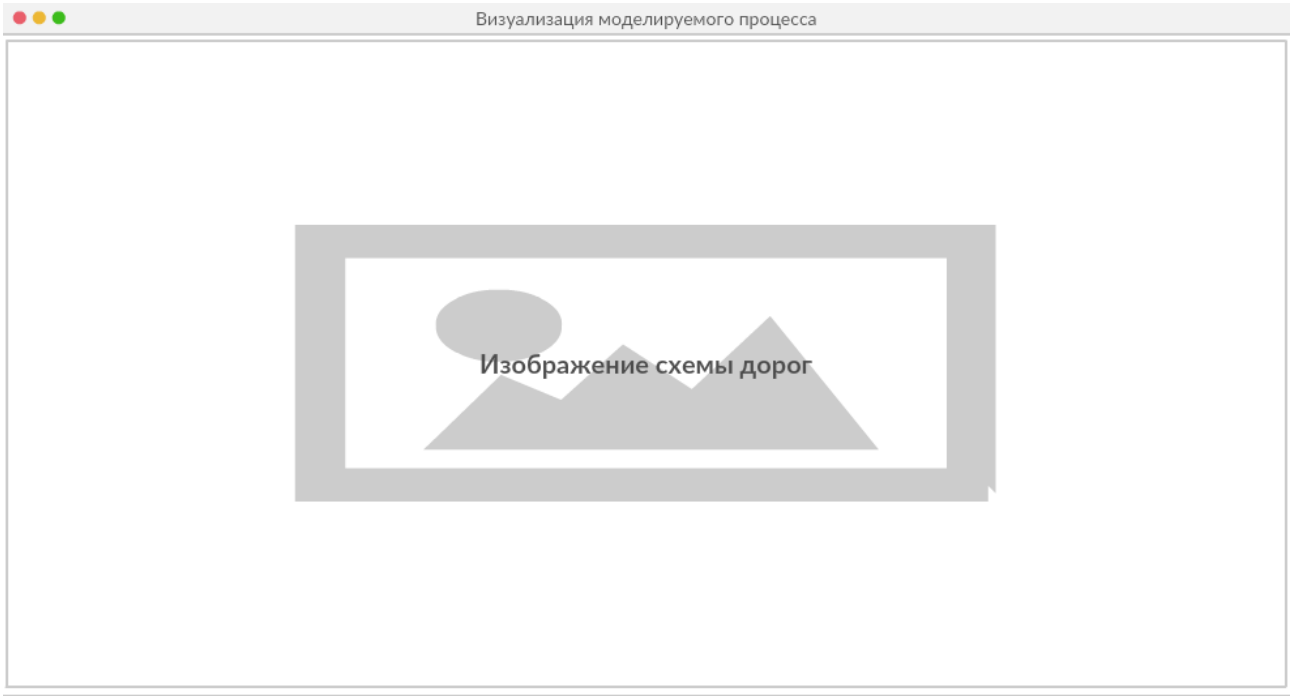

Рисунок 7 – Окно визуализации моделируемого процесса

Окно настройки маршрутов для движения транспорта (рисунок 8).

Данное окно должно содержать в себе таблицу, через которую пользователь сможет изменять данные о маршрутах и, при желании, сохранить изменения через специальную кнопку для этого. Окно также должно быть закрываемым.

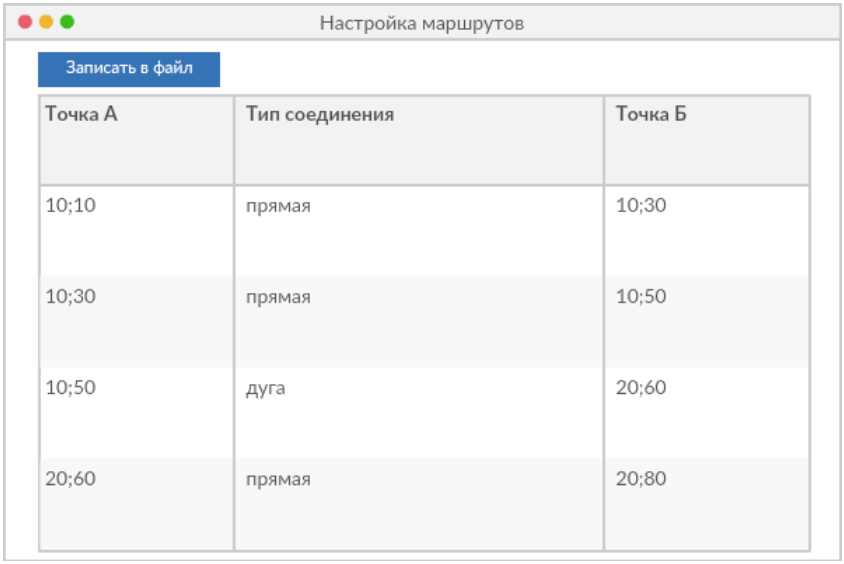

Рисунок 8 – Окно настройки маршрутов

Окно настройки параметров светофоров (рисунок 9).

Данное окно должно содержать в себе таблицу, через которую пользователь сможет изменять данные о светофорах и, при желании, сохранить изменения через специальную кнопку для этого. Окно также должно быть закрываемым.

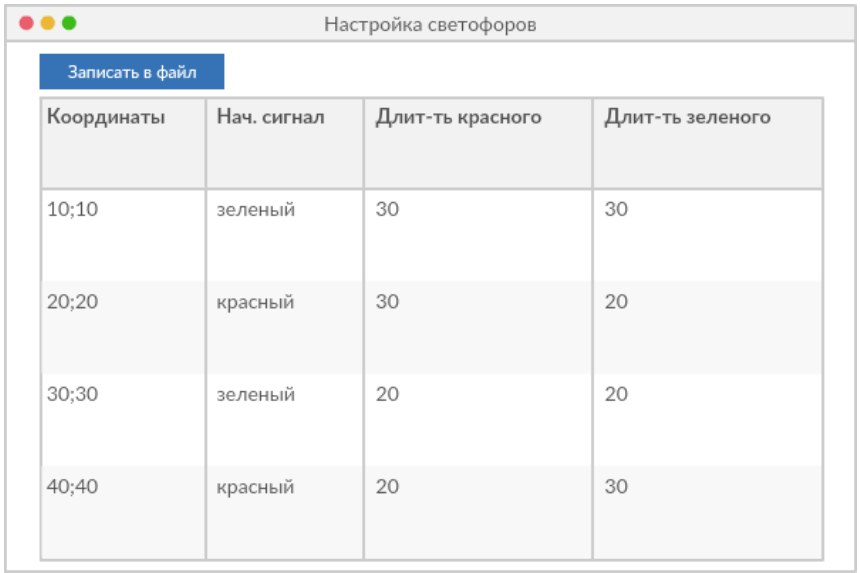

Рисунок 9 – Окно настройки светофоров

Программные интерфейсы:

Окно управления настройками модели должно считать все введенные и примененные пользователем данные и настройки при нажатии кнопки «Старт» и использовать данные настройки при визуализации моделирования. Также при нажатии кнопки «Старт» запускаются алгоритмы имитации движения и работы светофоров. Кнопка «Стоп» должна остановить алгоритмы и моделирование в целом. Панель переключения моделей при изменении состояния вызывает ряд функций, которые изменяют алгоритм работы светофоров и отображаемую модель. Закрытие окна освобождает память и завершает работу приложения.

Окно визуализации моделируемого процесса лишь обновляет свое состояние через системную функцию, отвечающую за это, когда она вызывается.

Окно настроек маршрутов при изменении данных передает их в соответствующие функции-обработчики, которые выполняют данную операцию напрямую с хранимыми данными. А при нажатии клавиши «Записать в файл» записывает все данные в файл, который хранит данные даже после завершения работы программы.

Окно настроек светофоров, как и в случае с окном настроек маршрутов, при изменении данных передает их в соответствующие функции-обработчики, которые выполняют данную операцию напрямую с хранимыми данными. А при нажатии клавиши «Записать в файл» записывает все данные в файл, который хранит данные даже после завершения работы программы.

Интерфейсы оборудования:

Данный продукт является чисто программным решением и из аппаратной части требуются лишь стандартные HID-устройства для ОС Windows: клавиатура, мышь, монитор.

Интерфейсы связи и коммуникации:

Программный продукт не предусматривает какие-либо коммуникации, за исключением тех, что заложены внутри ОС Windows.

2.5 Нефункциональные требования

Требования к производительности:

Предъявляются следующие требования к производительности:

- нагрузка на CPU не должна превышать 70%;

- использование физической/виртуальной памяти не должно превышать 128 МБ;

- обращения к базе данных должны осуществляться с частотой, задаваемой пользователем и время операции выполнения не превышать 1 секунды;

- время выполнения приложения неограниченно и контролируется пользователем;

- дисковые операции должны осуществляться с частотой, задаваемой пользователем;

- время отклика приложения не должно превышать 1 секунды;

- сборка мусора в коде не должна превышать 1 операции в секунду;

- время запуска приложения не должно превышать 1 секунды.

Требования к сохранности (данных):

Строгие требования к сохранности данных не предъявляются, поскольку программный продукт не хранит никакой информации, за исключением настроек, внесенных пользователем.

Обеспечить сохранность данных позволяет совокупность следующих способов:

- содержание данных или отдельных групп данных (нельзя читать информацию по отдельным объектам);

- ограничение доступа к функциям;

- применение шаблонных настроек приложения в случае отсутствия или утраты пользовательских.

Критерии качества программного обеспечения:

Модель качества программного обеспечения описана в стандарте ГОСТ Р ИСО/МЭК 9126-93 «Информационная технология. Оценка программной

продукции. Характеристики качества и руководства по их применению» и категоризирует атрибуты качества программного обеспечения по шести характеристикам: функциональные возможности, надежность, практичность, эффективность, сопровождаемость и мобильность.

Требования к безопасности системы:

Строгие требования к безопасности системы не предъявляются, поскольку для программного продукта достаточно соблюдать принципы объектно-ориентированного подхода и обрабатывать исключения.

3 Характеристика обеспечивающих элементов объекта проектирования

В данном разделе приводятся выбранные инструменты разработки и система управления проектом.

3.1 Выбор инструментов разработки

В качестве инструментов разработки выбраны среда Microsoft Visual Studio и язык программирования C#.

Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом [16].

В Visual Studio 2019 возможно создание приложений Windows с помощью Windows Presentation Foundation и Windows Forms.

В него входят следующие компоненты [17]:

- средства разработки классических приложений .NET;
- средства разработки для .NET Framework 4.x;
- средства профилирования .NET;
- поддержка языков C# и Visual Basic;
- инструменты для Entity Framework 6;
- IntelliTrace;
- JIT-отладчик;
- Live Unit Testing;
- Live Share.

Также в Visual Studio 2019 присутствуют ряд полезных инструментов для разработчика [17]:

- редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может

работать как отладчик уровня исходного кода, так и как отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (например, Subversion и VisualSourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода);

- CodeLens для получения важных аналитические сведений, например о внесенных в код изменениях, их последствиях и модульном тестировании методов. Можно мгновенно просматривать ссылки, авторов, тесты, журнал фиксаций и другие ключевые данные;

- отладчик, который позволяет приостанавливать выполнение кода с помощью нужных методов и точек останова;

- работа с наборами тестов помогает упорядочивать данные, анализировать объем тестируемого кода и мгновенно видеть результаты. Продвинутые функции, тестирующие код прямо во время ввода, позволяют быстро узнавать последствия каждого вносимого изменения. Проверка на прохождение новых изменений существующих тестов.

C# — объектно-ориентированный язык программирования.

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML [18].

Основные достоинства C# [19]:

- каждый компонент имеет атрибуты, предоставляющие декларативные сведения о компоненте, а также встроенные элементы документации. C# предоставляет языковые конструкции, непосредственно поддерживающие такую концепцию работы. Благодаря этому C# отлично подходит для создания и применения программных компонентов;

- надежность и устойчивость приложений: сборка мусора автоматически освобождает память, занятую уничтоженными и неиспользуемыми объектами; обработка исключений предоставляет структурированный и расширяемый способ выявлять и обрабатывать ошибки; строгая типизация языка не позволяет обращаться к неинициализированным переменным, выходить за пределы индексируемых массивов или выполнять неконтролируемое приведение типов;

- единая система типов. Все типы C#, включая типы-примитивы, такие как int и double, наследуют от одного корневого типа object. Таким образом, все типы используют общий набор операций, и значения любого типа можно хранить, передавать и обрабатывать схожим образом. Кроме того, C# поддерживает пользовательские ссылочные типы и типы значений, позволяя как динамически выделять память для объектов, так и хранить упрощенные структуры в стеке;

- предназначен для управления версиями. Многие языки программирования обходят вниманием этот вопрос, и в результате программы на этих языках ломаются чаще, чем хотелось бы, при выходе новых версий зависимых библиотек. Вопросы управления версиями существенно повлияли на такие аспекты разработки C#, как раздельные модификаторы virtual и override, правила разрешения перегрузки методов и поддержка явного объявления членов интерфейса.

3.2 Система управления проектом

В качестве системы управления проектом выбрана облачная платформа Azure DevOps [20].

Azure предоставляет комплексное автоматизированное решение для DevOps, в состав которого входит встроенная система безопасности и средства мониторинга. Там можно выбрать любой способ разработки и эксплуатации приложений в облаке. Возможности разработки Azure DevOps интегрируются с любыми средствами на выбор разработчика. Также возможно объединение специалистов по разработке, IT-операциям и проектированию качества для обеспечения создания, тестирования, развертывания, мониторинга и администрирования приложений в облаке. Для приложений и рабочих нагрузок, в которых предъявляются требования к задержке, нормативные или пользовательские требования, Azure совместно с Azure Stack предоставляет согласованную гибридную среду DevOps [21].

DevOps – это методика, объединяющая специалистов, процессы и технологии, которые имеют отношение к разработке и ИТ, в пяти основных областях: планирование и отслеживание, разработка, сборка и тестирование, доставка, а также мониторинг и эксплуатация. Благодаря DevOps специалисты по разработке, ИТ-операциям, проектированию качества и безопасности могут тесно сотрудничать, объединяя методики, которые раньше были изолированы. Улучшенная координация и совместная работа между представителями этих дисциплин сокращает время от момента внесения изменений в систему до их внедрения в рабочую среду. Кроме того, эта методология обеспечивает соблюдение стандартов безопасности и надежности в процессе работы. Результат – более совершенные продукты, которые доставляются быстрее, повышая уровень удовлетворенности клиентов [21].

Azure DevOps имеет определенные этапы и инструменты разработки (таблица 2) [21].

Таблица 2 – Этапы разработки Azure DevOps

Название этапа	Описание этапа
Планирование и отслеживание	Идентификация и отслеживание работы при использовании таких методик и процессов, как канбан-доски и Agile.
Разработка	Написание кода с использованием современных систем управления версиями, таких как Git, для безопасной непрерывной интеграции с главной ветвью.
Сборка и тестирование	Записывание кода после изменения в Git или другую систему управления версиями запускает процесс автоматической сборки. Код проходит тестирование и проверку — ошибки устраняются на раннем этапе разработки, пока их исправление менее затратно. Процесс автоматической сборки и тестирования называется непрерывной интеграцией (CI). Артефакт, готовый к развертыванию в рабочей среде, – это результат успешной сборки и интеграции, которые создают условия для непрерывной поставки (CD).
Развертывание	После тестирования и проверки каждое изменение развертывается в рабочей среде. Благодаря методике непрерывной поставки финальное развертывание в рабочей среде представляет собой бизнес-решение, которым можно управлять вручную. Непрерывное развертывание позволяет автоматизировать весь процесс – от фиксации кода до публикации в рабочей среде. При автоматическом развертывании кода клиенты получат доступ к новым компонентам, как только они будут готовы.
Мониторинг и эксплуатация	В рабочей среде мониторинг обеспечивает поступление информации о производительности и шаблонах использования приложения. Возможность устранять неполадки и собирать данные помогает принимать взвешенные бизнес-решения, касающиеся дальнейшей разработки. А благодаря автоматически применяемым политикам соответствия, в приложениях, развернутых в рабочей среде, будут применяться конфигурации требуемого состояния, соответствующие рекомендациям по обеспечению безопасности.

В качестве подхода к разработке программного обеспечения выбрана методология Agile.

В качестве системы контроля версий выбран Team Foundation Version Control для разработки и последующий переход на GitHub на этапе релиза.

В качестве инструмента тестирования выбран встроенный в Visual Studio инструмент Microsoft Test Framework.

4 Процессы проектирования программного обеспечения

В данном разделе рассматриваются такие этапы проектирования, как планирование этапов разработки, проектирование архитектуры, формирование описание алгоритмов, тестирование, проектирование интерфейсов и описание информационно-логической модели базы данных.

4.1 Этапы разработки программного продукта

Программный продукт разрабатывался в два этапа, у которых есть спринты (таблица 3). Первый этап был завершен в конце 2018 года и представлял из себя программную модель имитации дорожного движения автомобилей. Второй этап начался в начале 2019 года и был завершен в начале июля этого же года: в него входила разработка алгоритмов регулирования светофором на разработанной ранее программной модели.

Таблица 3 – Спринты разработки проекта

Название спринта	Сроки
1 этап	
Инициализация проекта	13.09.2018 – 30.09.2018
Разработка модели дорог	01.10.2018 – 31.10.2018
Описание алгоритмов движения автомобилей	01.11.2018 – 30.12.2018
2 этап	
Описание алгоритма обычного светофора	01.01.2019 – 31.01.2019
Описание алгоритма адаптивного светофора	01.02.2019 – 31.03.2019
Описание алгоритма взаимодействующего адаптивного светофора	01.04.2019 – 31.05.2019
Тестирование проекта	01.06.2019 – 31.06.2019
Завершение проекта	01.07.2019 – 07.07.2019

4.2 Проектирование архитектуры программного продукта

Разрабатываемый продукт представлен в виде решения Visual Studio, в который входит проект приложения Windows Forms с названием “PSMoTR” и проект модульного теста .Net Framework с названием “PSMoTRTests”.

Рассмотрим архитектуру проекта “PSMoTR” (рисунок 10). Здесь можно выделить три ключевых пакета: пакет с формами “Forms” (окнами Windows-приложения), пакет с классами “Classes” и пакет “Resources”, в который входят все используемые ресурсы проекта.

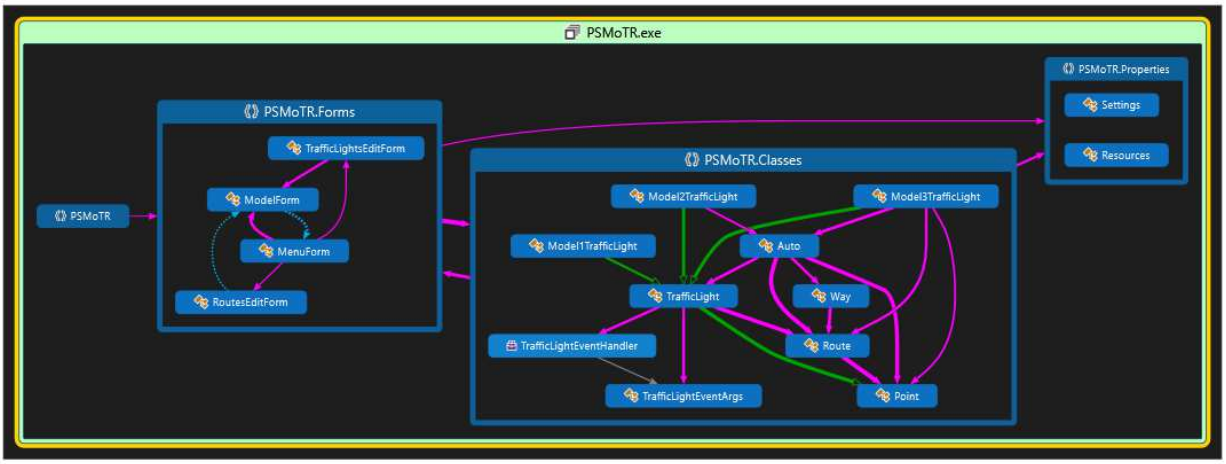

Рисунок 10 – Общая структура проекта “PSMOTR”

Рассмотрим структуру пакета “Classes” (рисунок 11). В нем описаны следующие классы:

- “Point” – класс объектов, представляющих точку на модели;

- “Route” – класс объектов, представляющих отрезки, по которым двигаются автомобили;

- “Auto” – класс объектов-автомобилей;

- “Way” – класс объектов, включающий набор объектов-отрезов “Route”, которые формируют путь автомобилей;

- “TrafficLight” – класс объектов, представляющих светофоры. Также он является базовым классом для трех моделей светофоров.

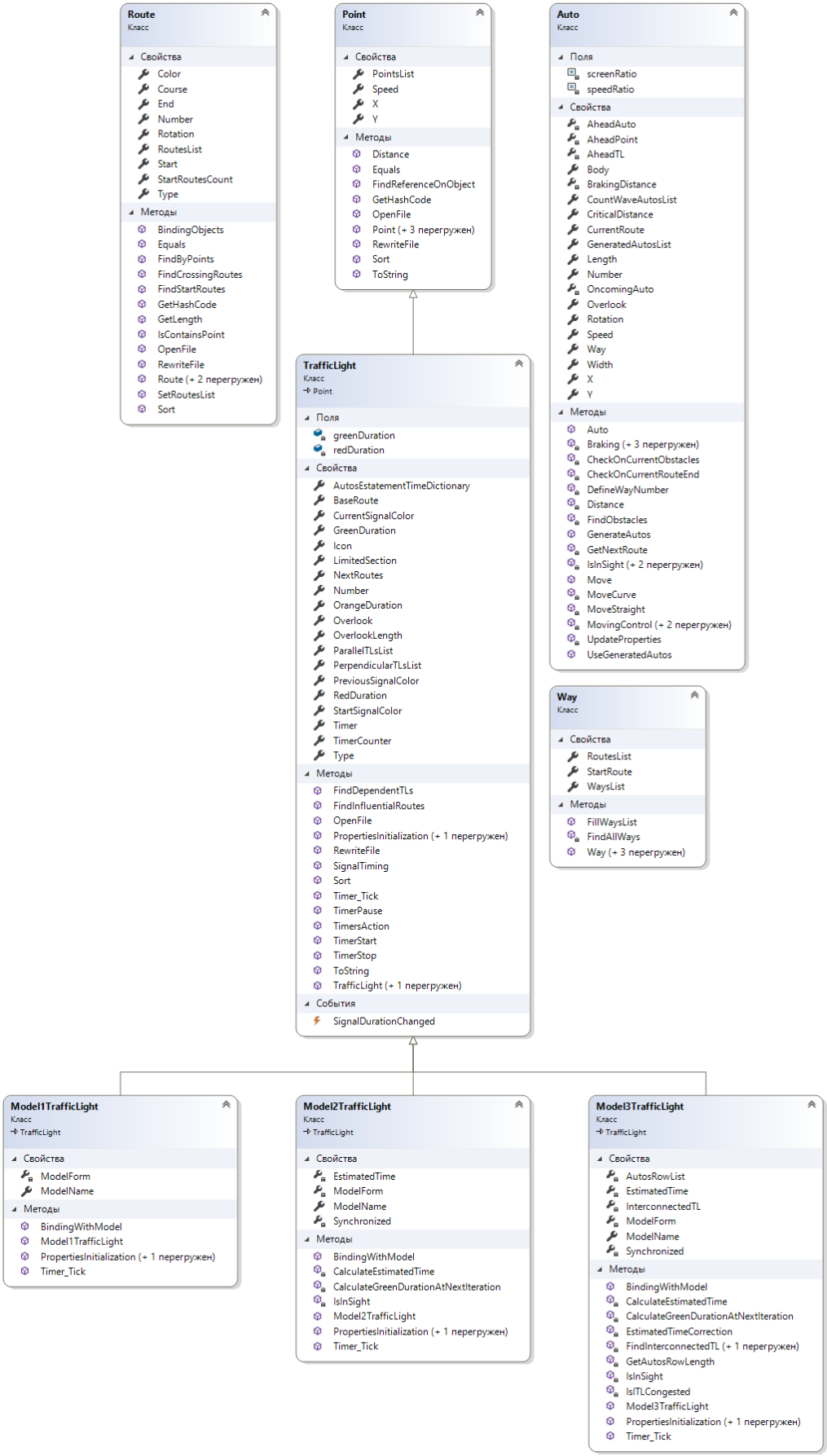

Рисунок 11 – UML-диаграмма пакета “Classes” проекта “PSMoTR”

Рассмотрим структуру пакета форм “Forms” (рисунок 12). В нем описаны следующие формы:

- “MenuForm” – форма для настройки модели;
- “ModelForm” – форма отображения модели;
- “RoutesEditForm” – форма редактирования маршрутов;
- “TrafficLightEditForm” форма редактирования светофоров.

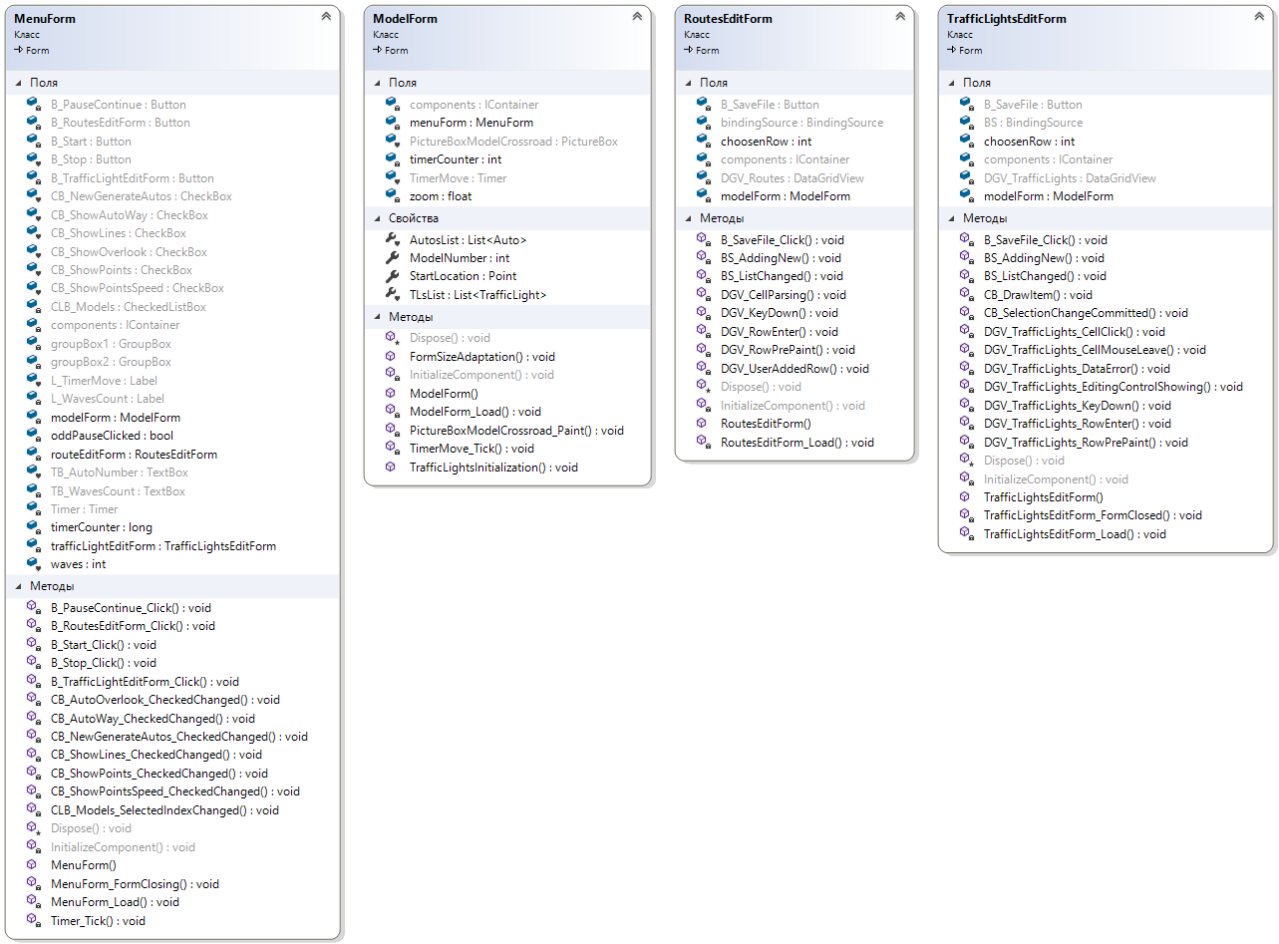

Рисунок 12 - UML-диаграмма пакета “Forms” проекта “PSMoTR”

Пакет “Resources” является хранилищем ресурсов проекта, таких как файлы и картинки. Однако сами ресурсы добавляются через свойства проекта “Properties.Resources.resx”, который и формирует данный пакет.

Помимо этого, в проекте есть файл конфигурации “App.config”, используемые библиотеки в разделе «Ссылки» и настройки пользователя/приложения в “Properties.settings”.

4.3 Описание алгоритмов программного продукта

В программном продукте модель имитации запускается и останавливается пользователем. При старте модели запускаются таймеры классов и объектов, которые периодически вызывают определенные методы:

- поиск всех возможных путей;
- генерация пути автомобиля;
- генерация автомобилей (количество генераций задается пользователем);
- методы движения автомобилей;
- методы переключения сигналов светофора;
- перерисовка модели.

Помимо данных действий, также существуют алгоритмы при работе с редакторами светофоров и маршрутов и считывании данных:

- связывание концевых точек маршрутов в случае совпадения координат;
- поиск пересекающих маршрутов;
- поиск зависимых светофоров для текущего.

Рассмотрим алгоритм поиска всех возможных путей (рисунок 13). Он рекурсивный и принимает на вход параметр “way” (в который записывается список маршрутов) и “currentRoute” – маршрут, от которого осуществляется поиск других маршрутов. Здесь в алгоритме возможны три исхода: маршрут найден один, маршрутов найдено более одного и маршрутов более не найдено. В первых двух случаях осуществляется рекурсия, а в последнем записывается путь, который сформировался до достижения данного этапа. Таким образом формируется список уникальных путей по существующим маршрутам.

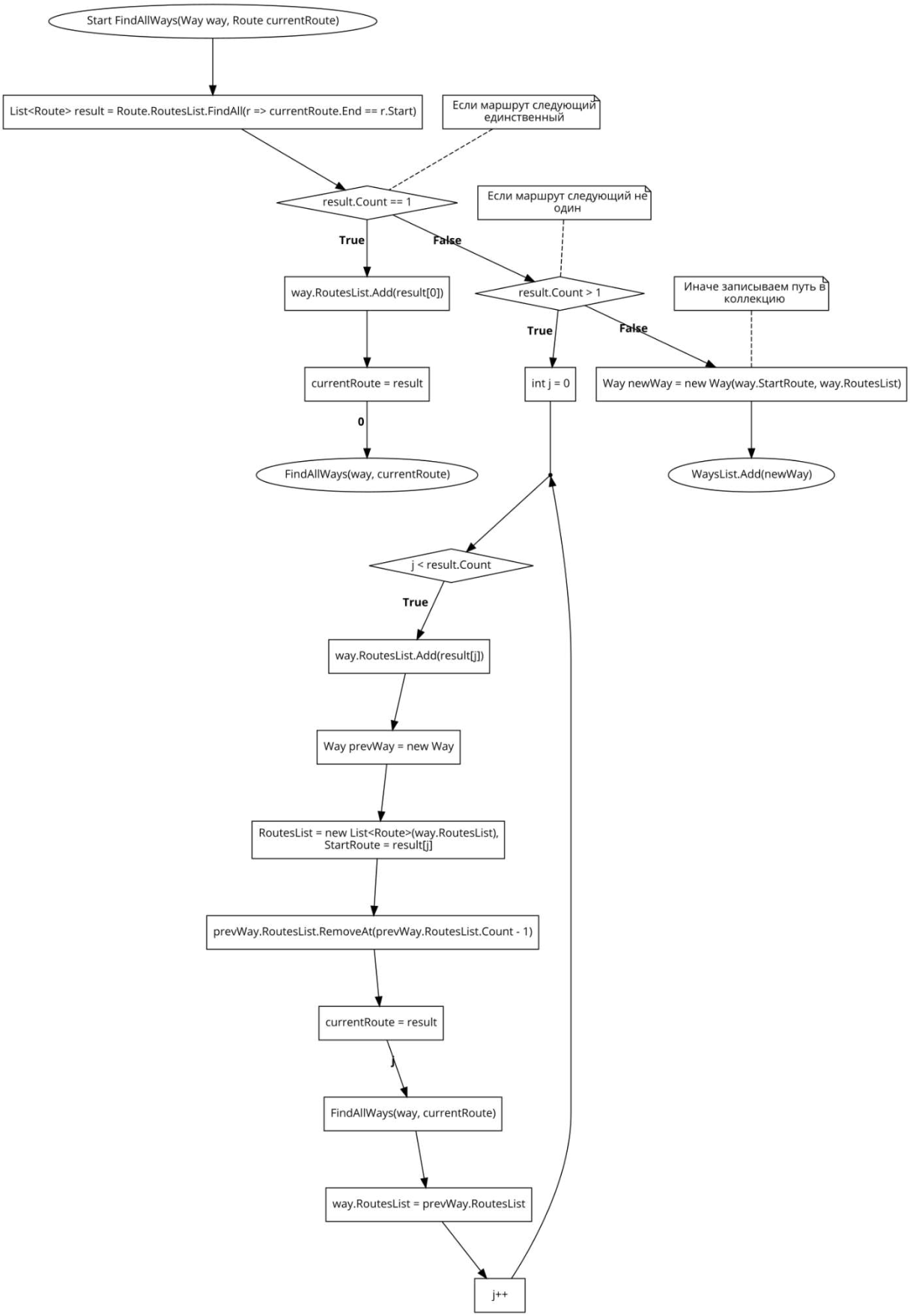

Рисунок 13 – Алгоритм поиска всех возможных путей

Рассмотрим алгоритм генерации пути автомобиля (рисунок 14). В него передаются начальное и конечное значение номеров автомобилей в списке, а также сам список автомобилей. В нем случайно выбирается путь через генератор случайных чисел и далее проверяется, не имеет ли уже автомобиль пути и уникален ли путь среди автомобилей выбранного диапазона. Возвращает в итоге номер пути.

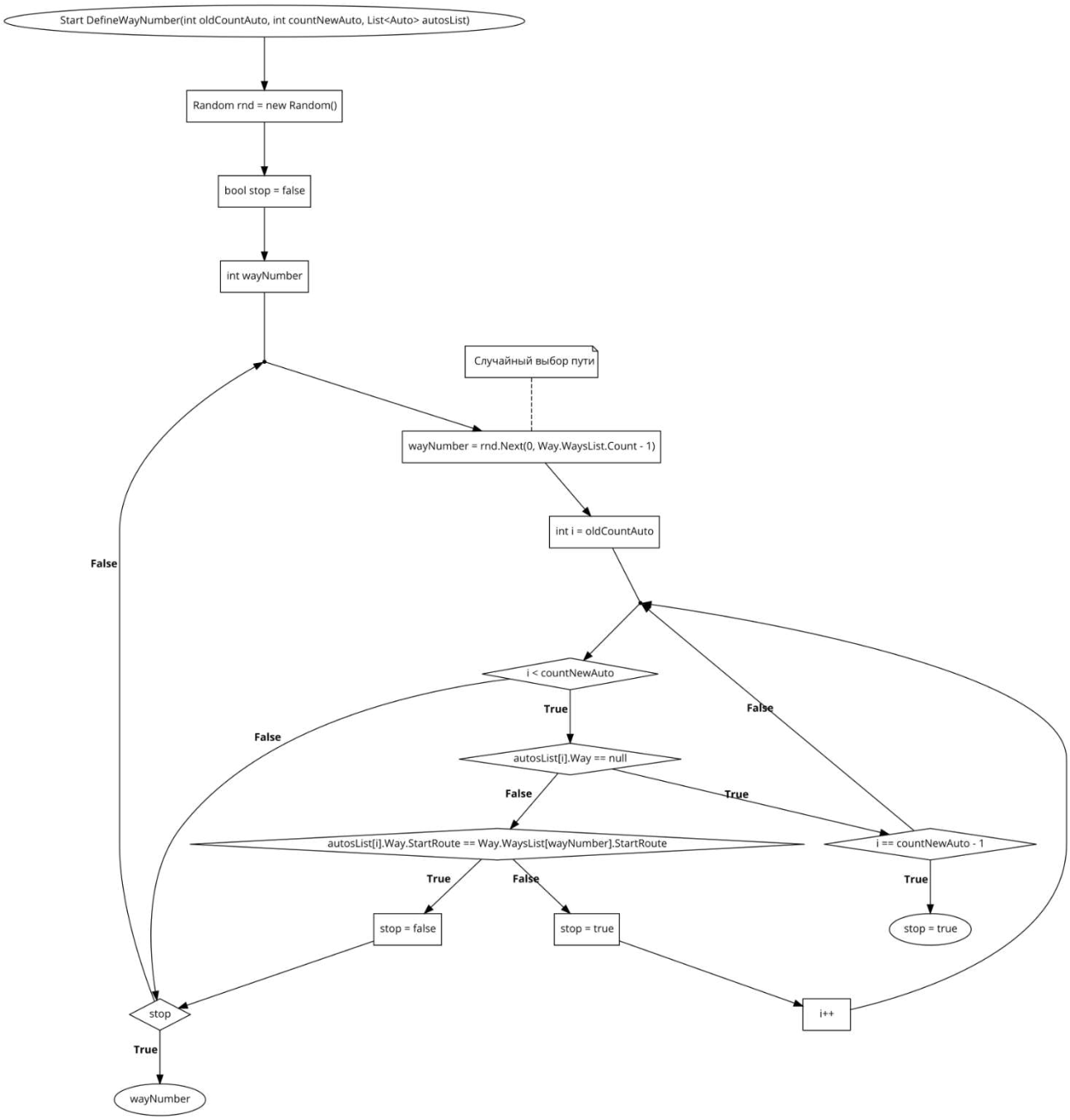

Рисунок 14 – Алгоритм генерации пути автомобиля

Рассмотрим алгоритм генерации автомобилей (рисунок 15) и используемые методы для движения автомобилей (рисунок 16).

Генерация автомобилей с помощью счетчика случайных чисел определяет, сколько автомобилей будет сгенерировано (от 1 до максимального количества стартовых маршрутов). Далее определяется диапазон в списке для новых авто и производится в цикле создание объектов авто и, при помощи метода генерации пути автомобиля, закладываем в данные объекты пути для движения.

Далее при запуске пользователем модели производится запуск таймера, у которого событие тика вызывает у каждого объекта автомобиля метод “Move”, который в свою очередь вызывает следующие методы:

- “FindObstacles” – метод, отвечающий за поиск помех для автомобиля;
- “CheckOnCurrentObstacles” – метод, отвечающий за поведение автомобиля для найденных помех (вызов метода “MovingControl” для каждого типа помехи);
- “CheckOnCurrentRouteEnd” – метод, отвечающий за проверку достижения конца текущего маршрута автомобиля и, в случае достижения, смену маршрута согласно пути;
- “MoveStraight” – метод, выполняющий изменение координат и угла поворота автомобиля по прямой;
- “MoveCurve” – метод, выполняющий изменение координат и угла поворота автомобиля по дуге (отдельно описаны методы движения по дуге для оси X и для оси Y);
- “UpdateProperties” – метод, обновляющий свойства автомобиля после завершения всех предыдущих.

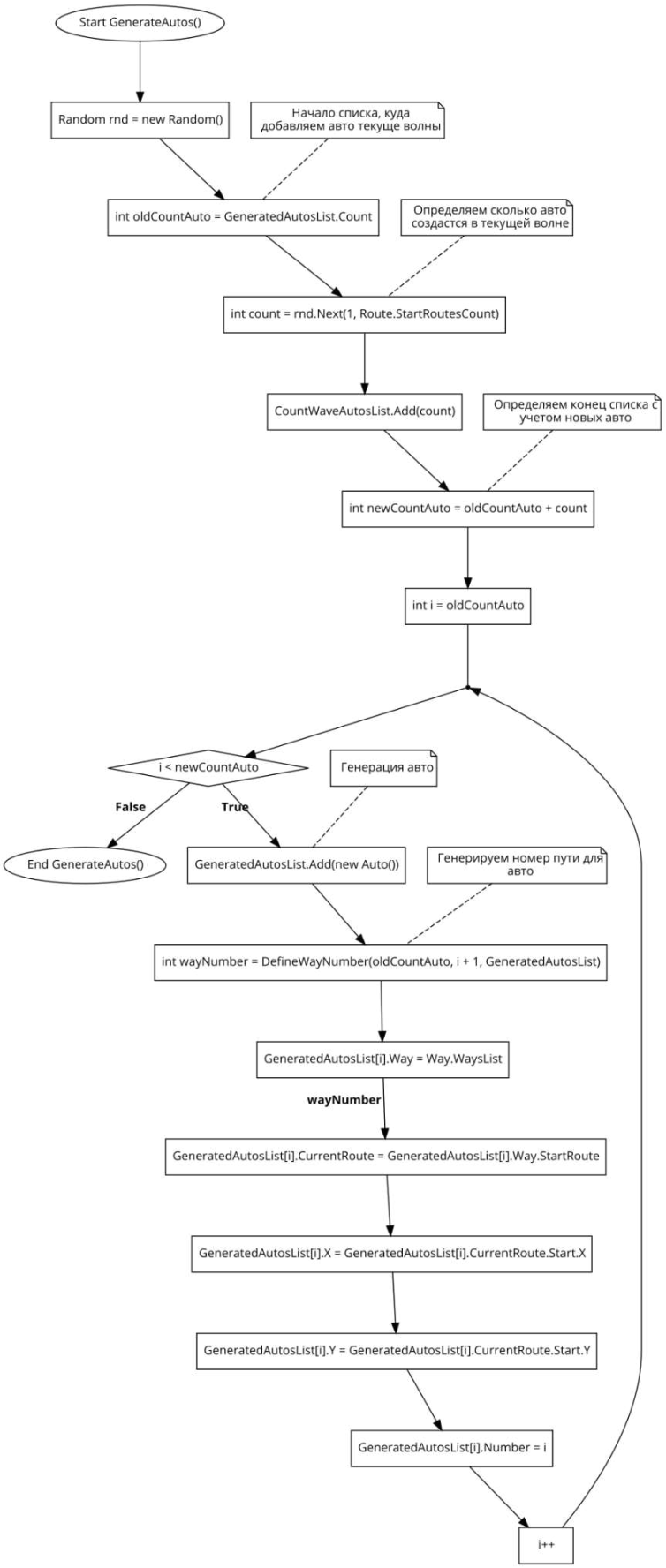

Рисунок 15 – Алгоритм генерации автомобилей

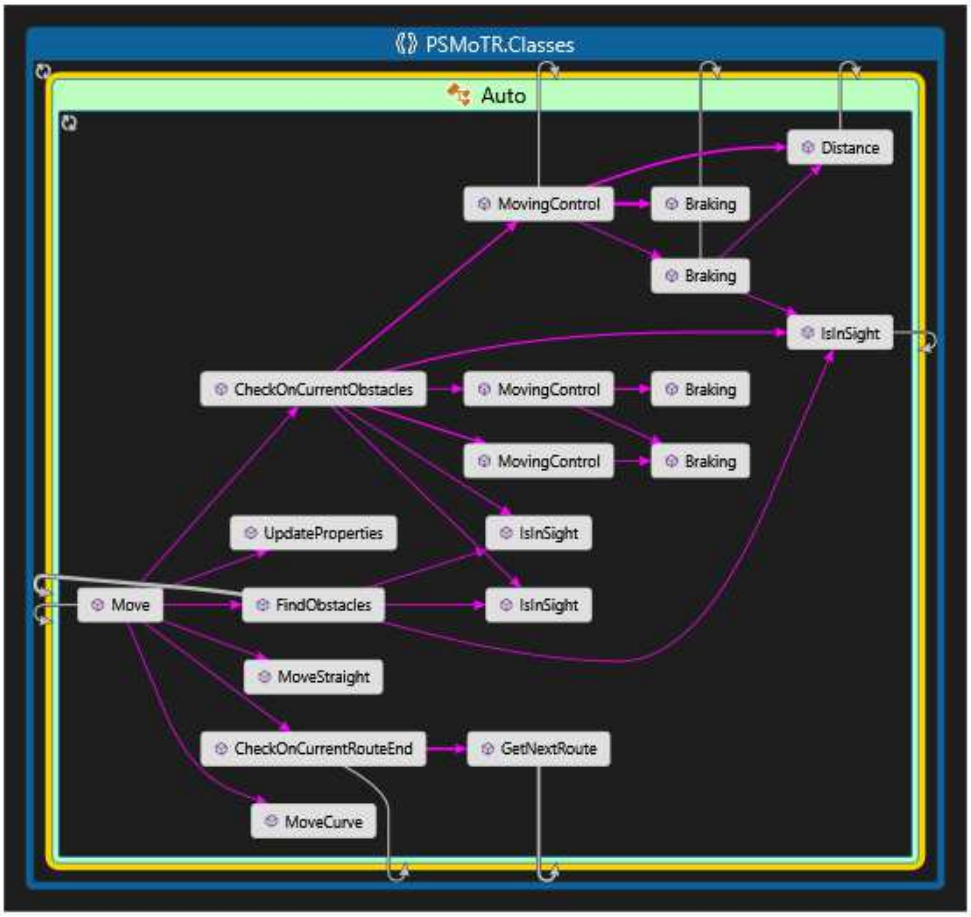

Рисунок 16 – Порядок вызова методов при движении автомобиля

Рассмотрим алгоритм связывания концевых точек маршрутов (рисунок 17). Данный алгоритм используется только при редактировании маршрутов и в случае, когда пользователь задает координату конца маршрута на существующую точку конца другого маршрута, то происходит «склеивание». В самом методе, отвечающим за добавление нового маршрута, перед вызовом данного метода осуществляется проверка ряда условий, которые не позволяют создавать дубликаты точек в списке точек в такой ситуации.

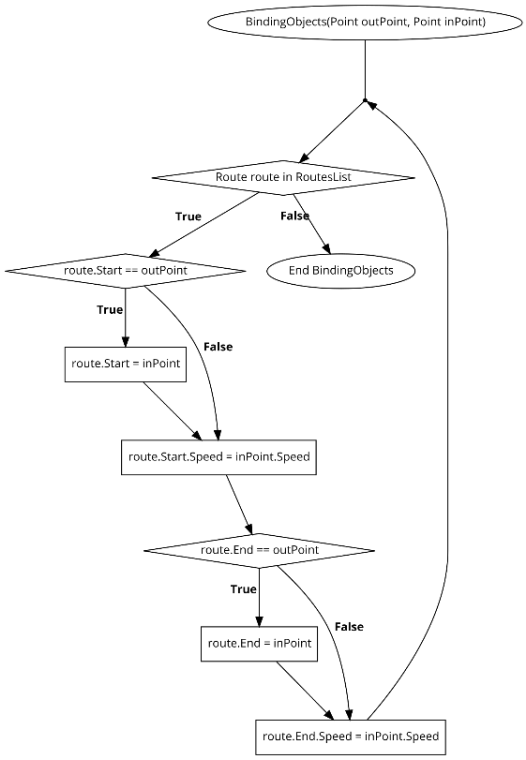

Рисунок 17 – Алгоритм связывания концевых точек маршрутов

Рассмотрим алгоритм поиска пересекающих маршрутов (рисунок 18). Данный алгоритм используется как при проверке движения автомобиля, когда он должен пересечь встречную полосу и проверить на наличие автомобилей там (то есть ищет автомобили, которые пересекают маршрут текущего автомобиля) и в алгоритме поиска зависимых светофоров для текущего светофора (рисунок 19). Данный алгоритм перебирает все существующие маршруты: приводит рассматриваемые прямые к единому направлению и осуществляет ряд проверок на то, какие они по отношению друг к другу (параллельные, перпендикулярные или иное). В зависимости от данных проверок осуществляется уже сопоставление координат точек и в итоге формируется список всех маршрутов, пересекающих текущий.

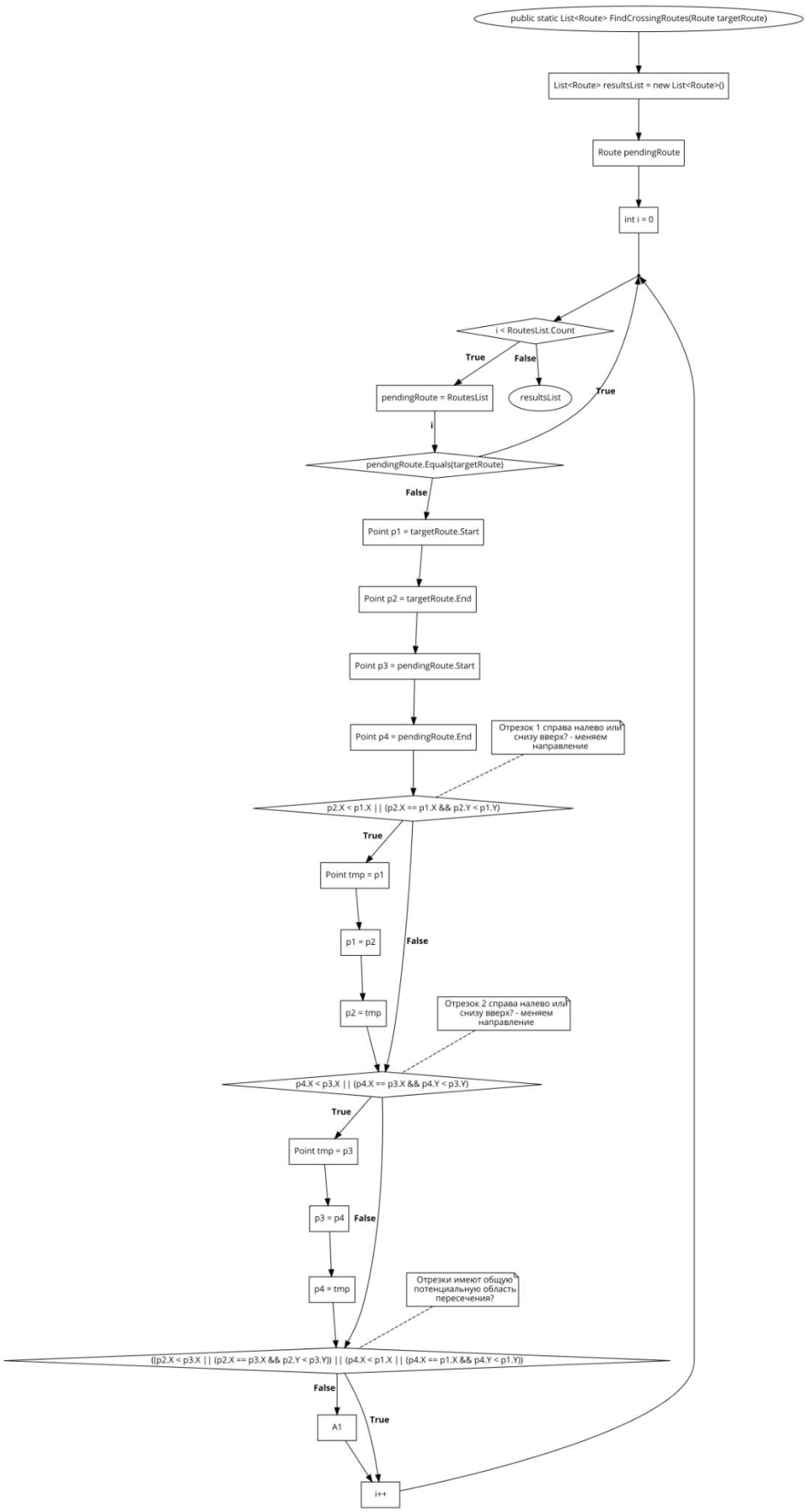

Рисунок 18 – Алгоритм поиска пересекающих маршрутов, лист 1

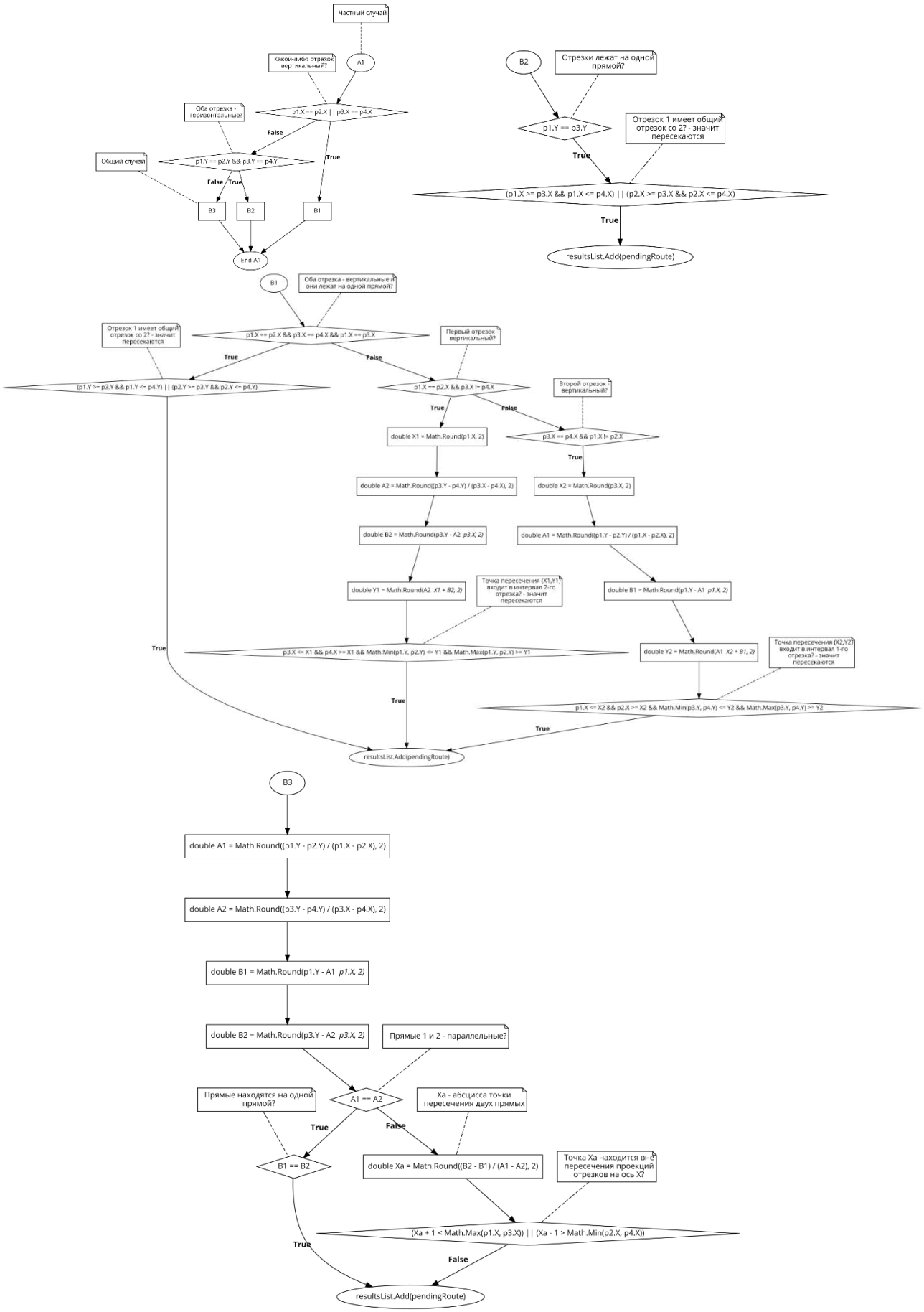

Рисунок 18, лист 2

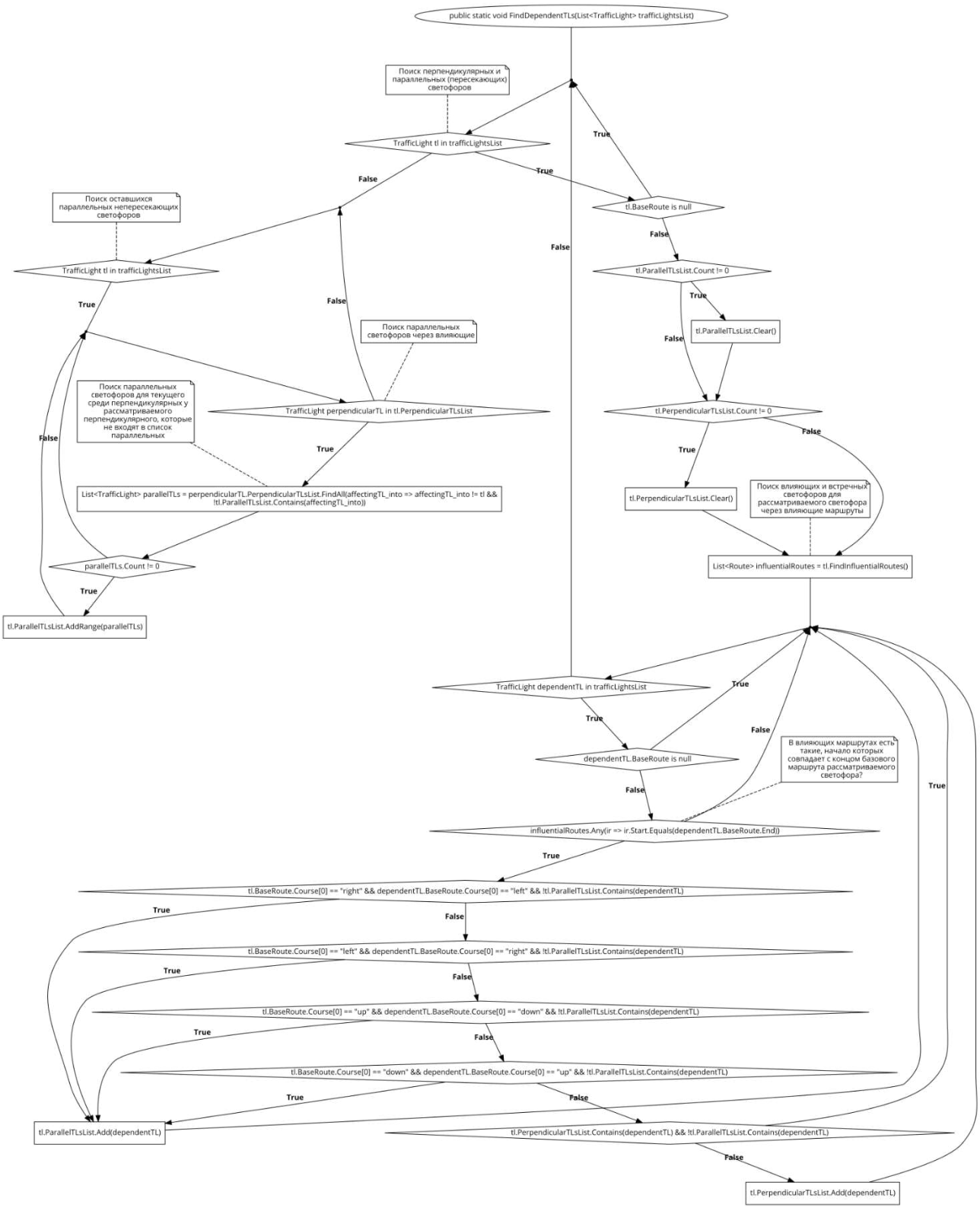

Рисунок 19 – Алгоритм поиска зависимых светофоров для текущего

Отдельно рассмотрим алгоритмы светофоров для каждой модели: модель 1 – ССУРС (рисунок 20), модель 2 – АСУРС (рисунок 21), модель 3 – ВАСУРС (рисунок 22).

Алгоритм ССУРС с каждым тиком таймера светофора осуществляет изменение и проверку счетчика таймера, в результате проверки которого осуществляется смена текущего сигнала и значения допустимой скорости для автомобилей, где “Speed = 0” используется совместно с красным сигналом и означает остановку движения для автомобилей.

Алгоритм АСУРС с каждым тиком таймера использует алгоритм ССУРС и дополнительно два метода для адаптации зеленого сигнала под колонну автомобилей перед светофором: “CalculateEstimatedTime” – вычисление оценочного времени по длине автоколонны перед светофором и “CalculateGreenDurationAtNextIteration” – выбор светофора для адаптации зеленого сигнала и смена значения зеленого сигнала в следующей итерации.

Алгоритм ВАСУРС использует те же методы в таймере тика, что и АСУРС, но модифицированные. В методе “CalculateEstimatedTime” помимо формирования оценочного времени ведется запись автоколонн перед вычисляемыми светофорами. А в методе “CalculateGreenDurationAtNextIteration” при выборе светофора для адаптации вызывается метод “EstimatedTimeCorrection”, который переоценивает будущую длительность зеленого сигнала, обращаясь у текущего светофора к зависимому светофору за информацией о том, сколько автомобилей перед ним и сколько есть свободного пространства на его участке дороги.

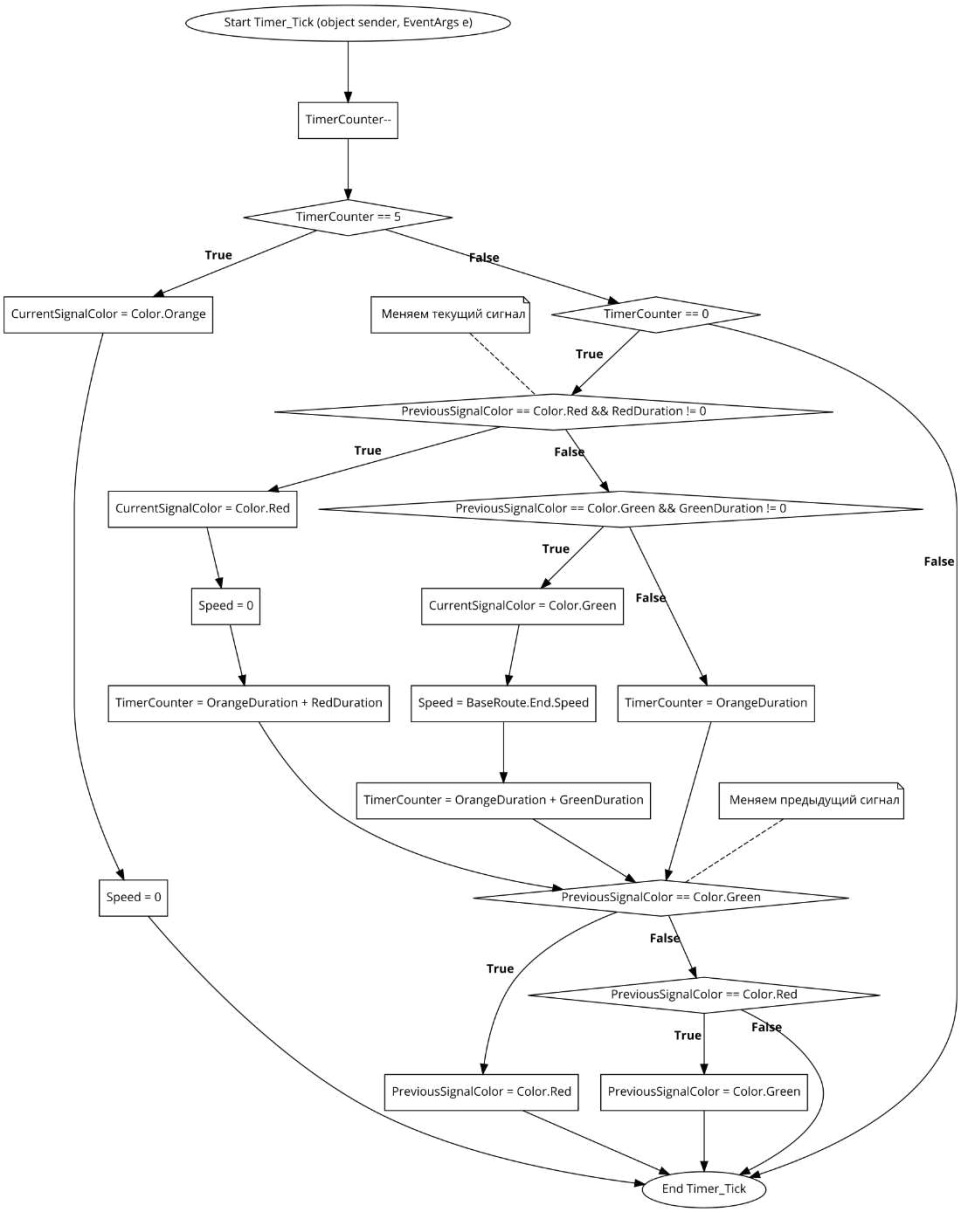

Рисунок 20 – Алгоритм ССУРС (модель 1)

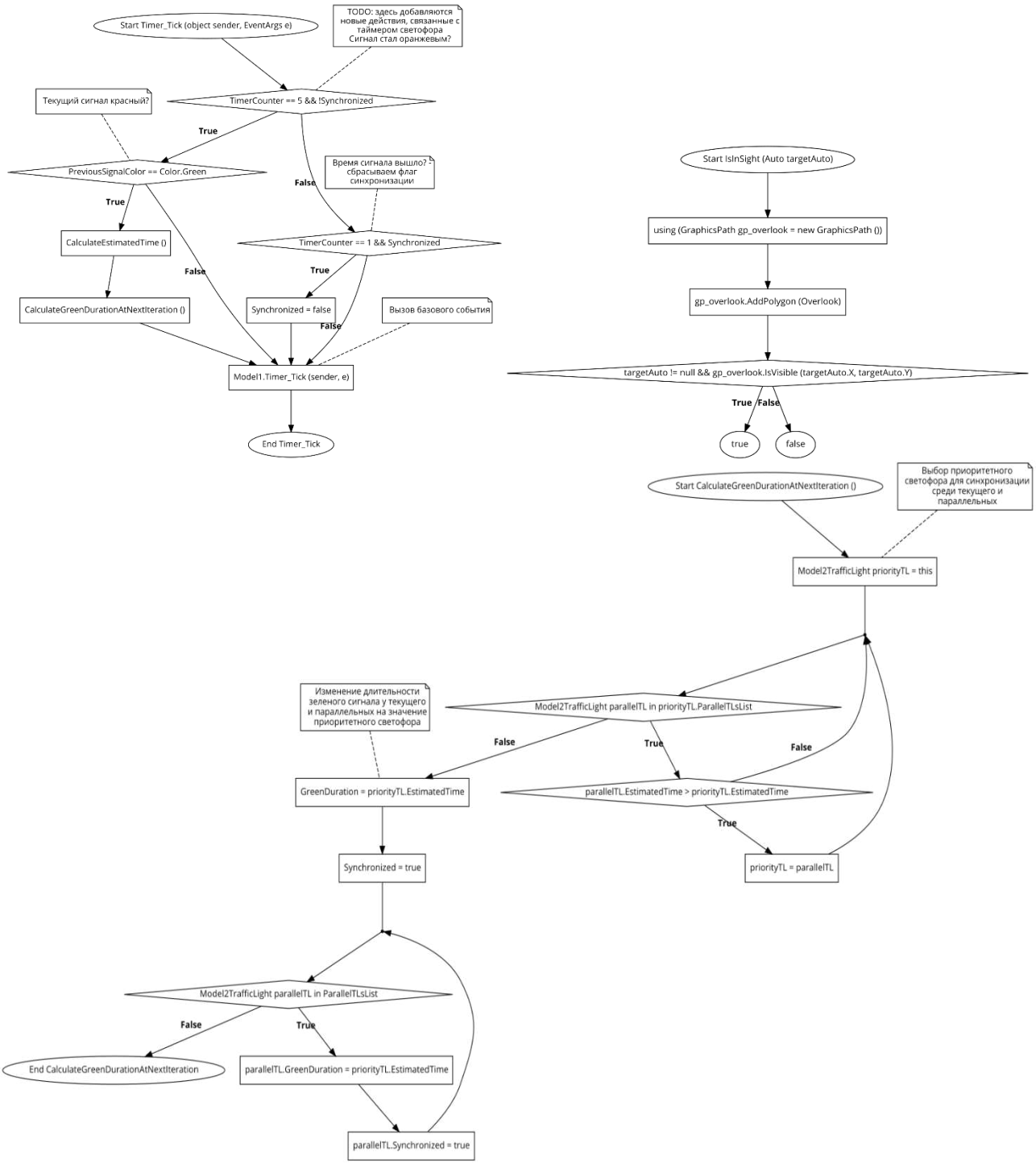

Рисунок 21 – Алгоритм АСУРС (модель 2), лист 1

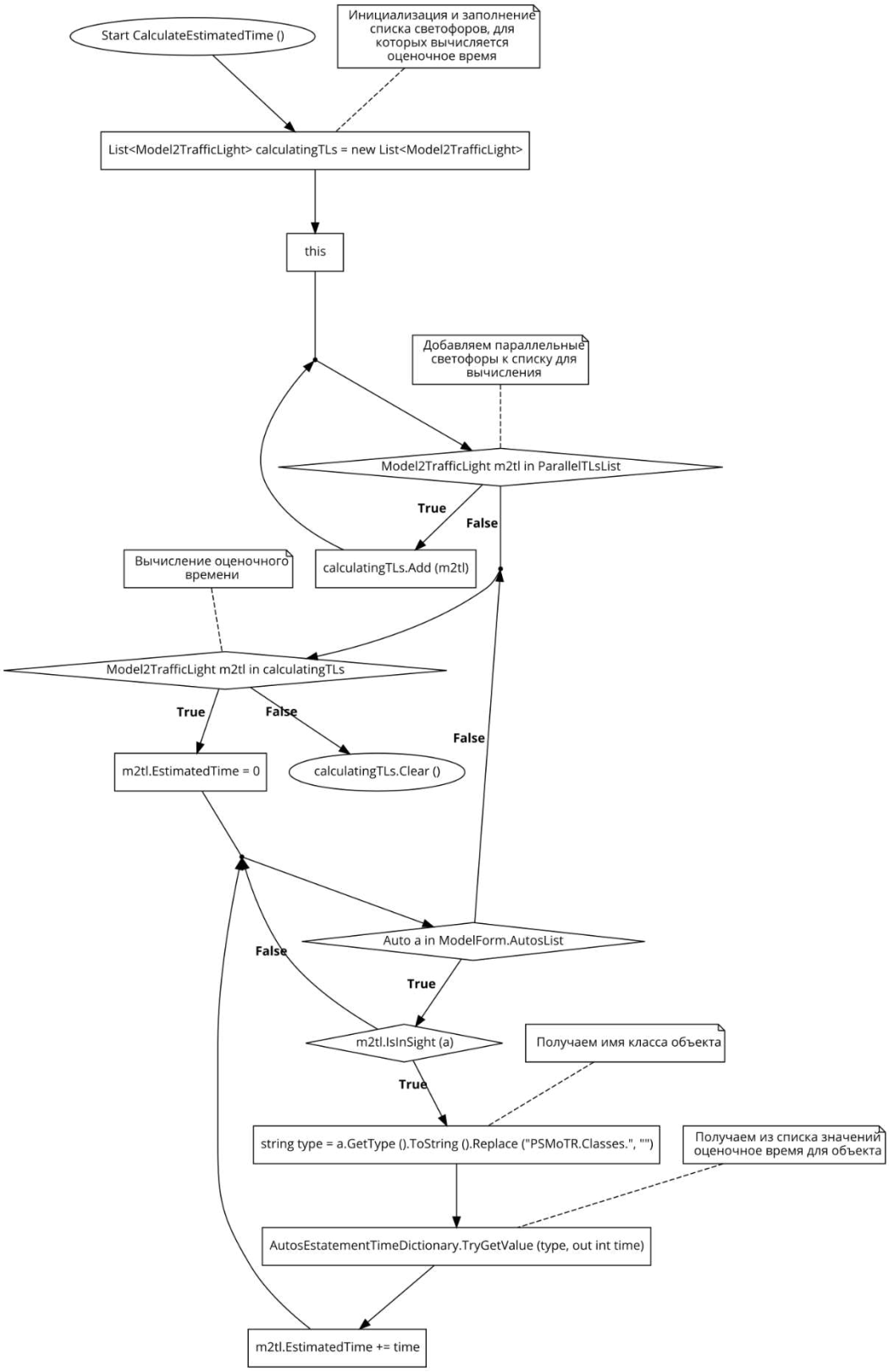

Рисунок 21, лист 2

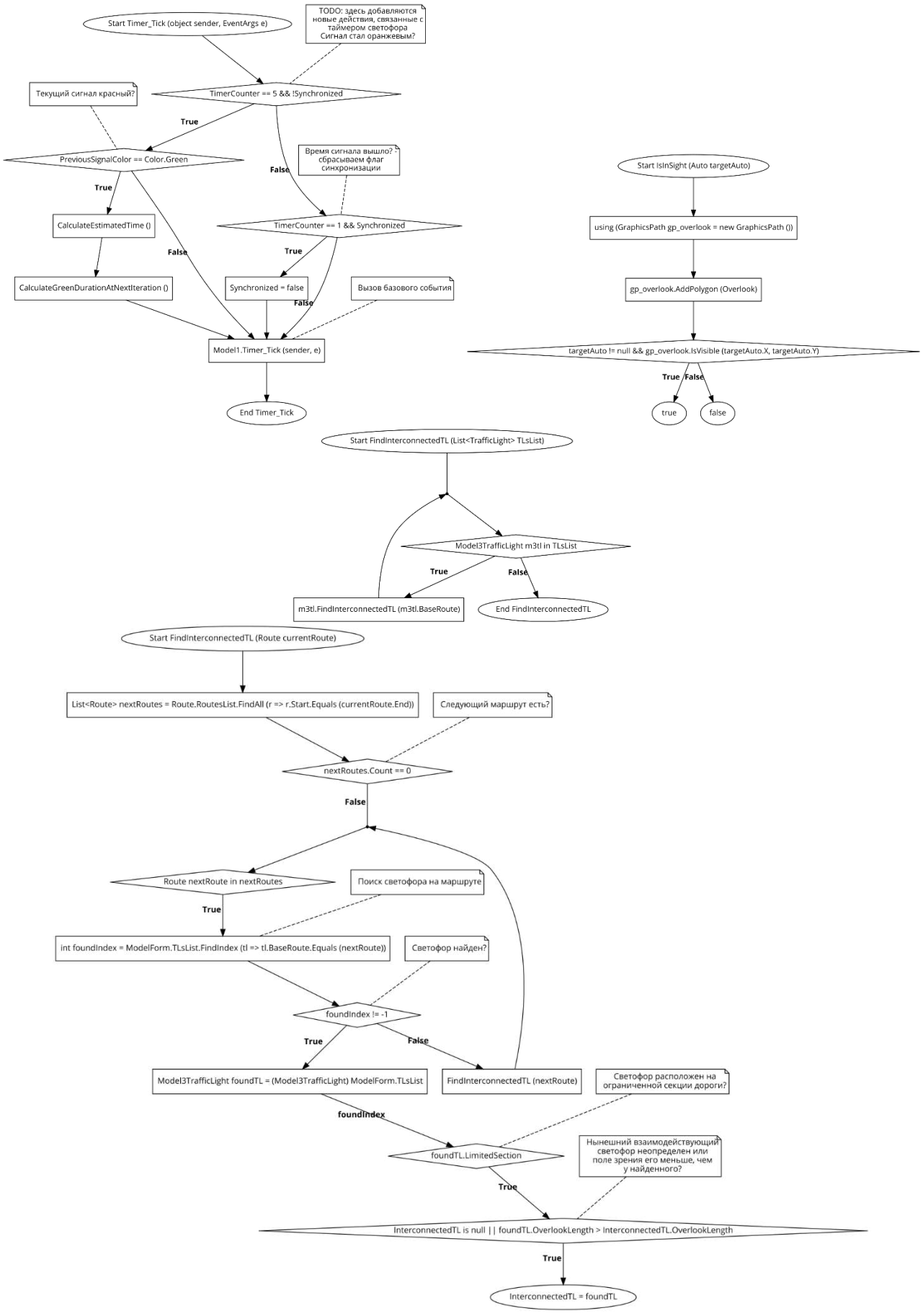

Рисунок 22 – Алгоритм ВАСУРС (модель 3), лист 1

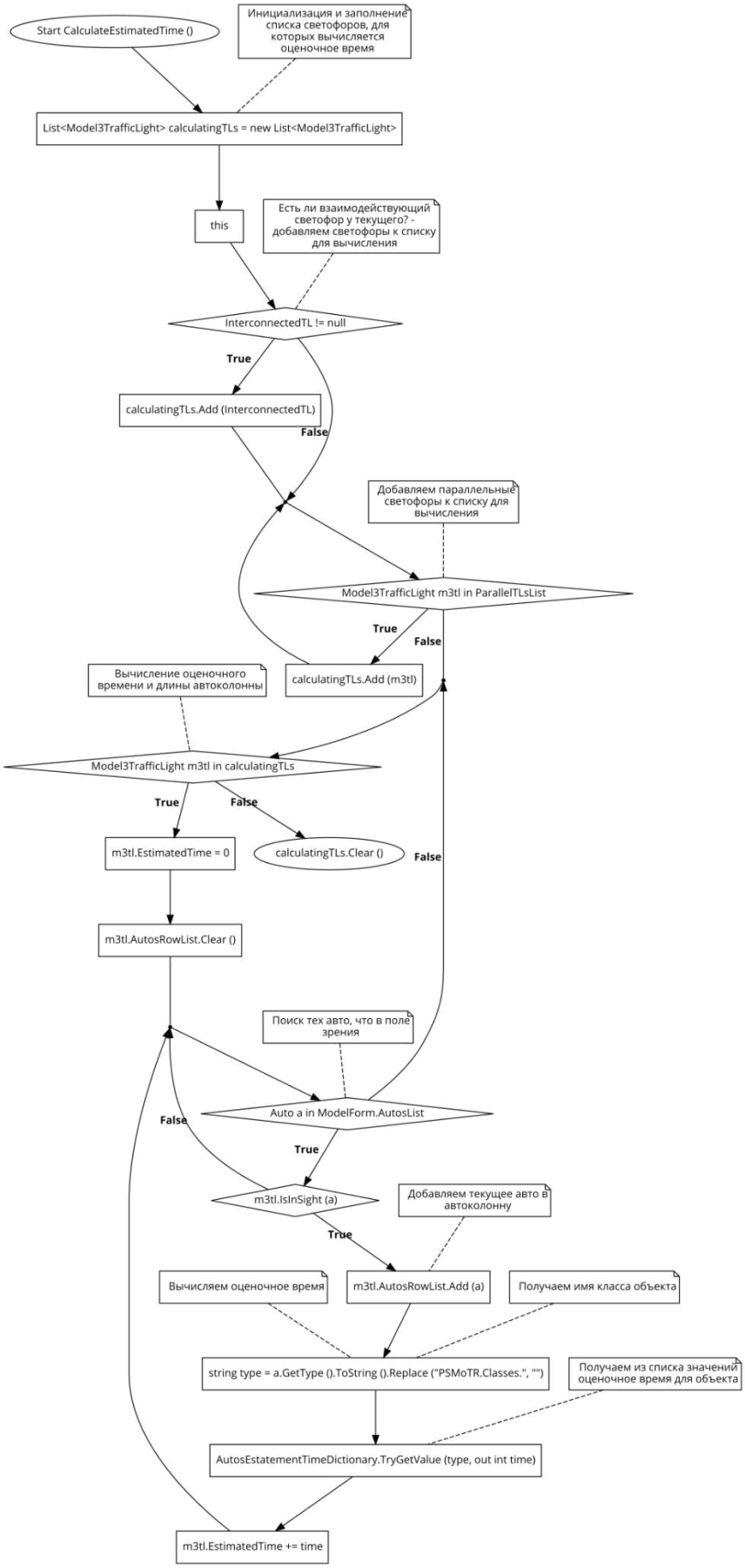

Рисунок 22, лист 2

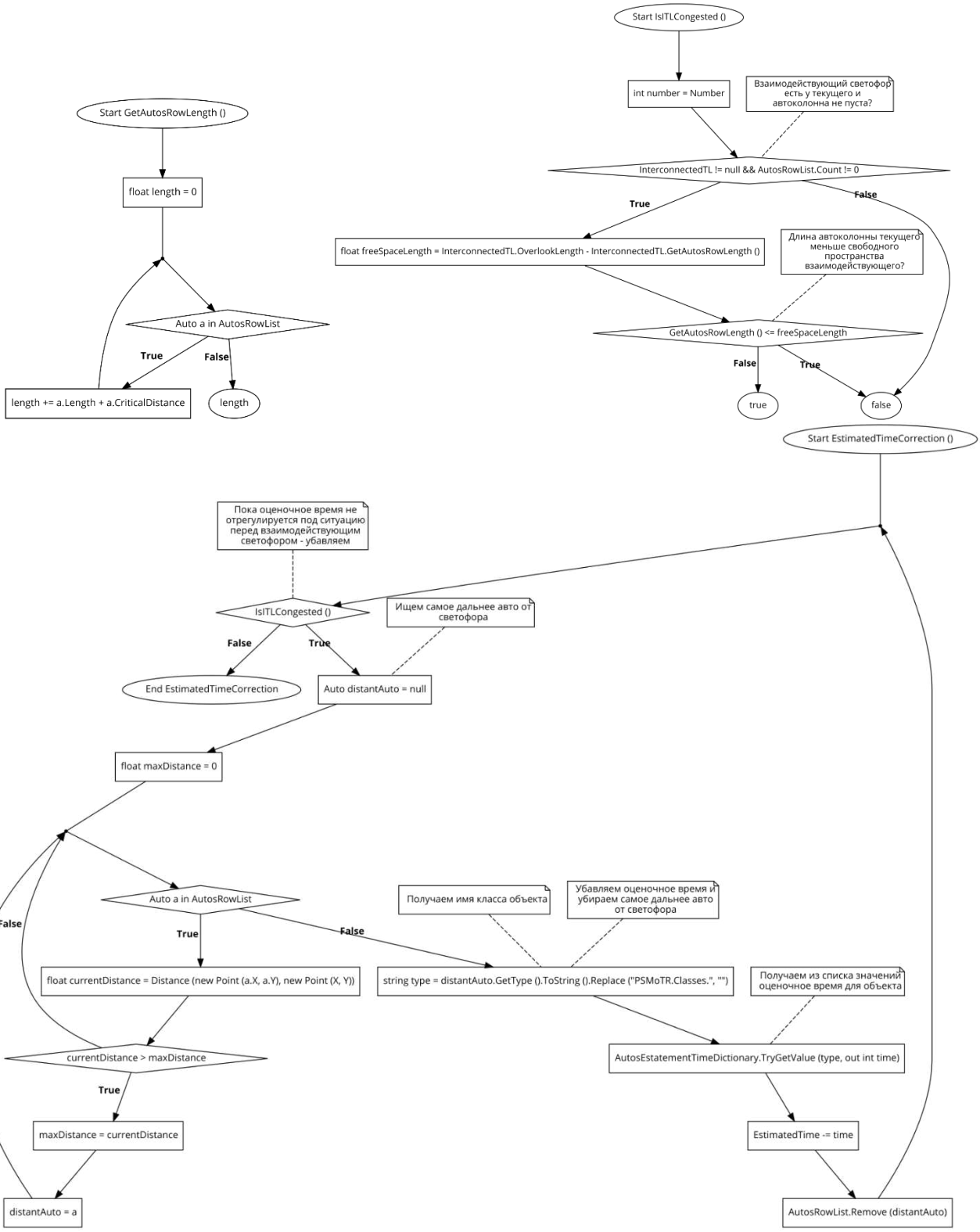

Рисунок 22, лист 3

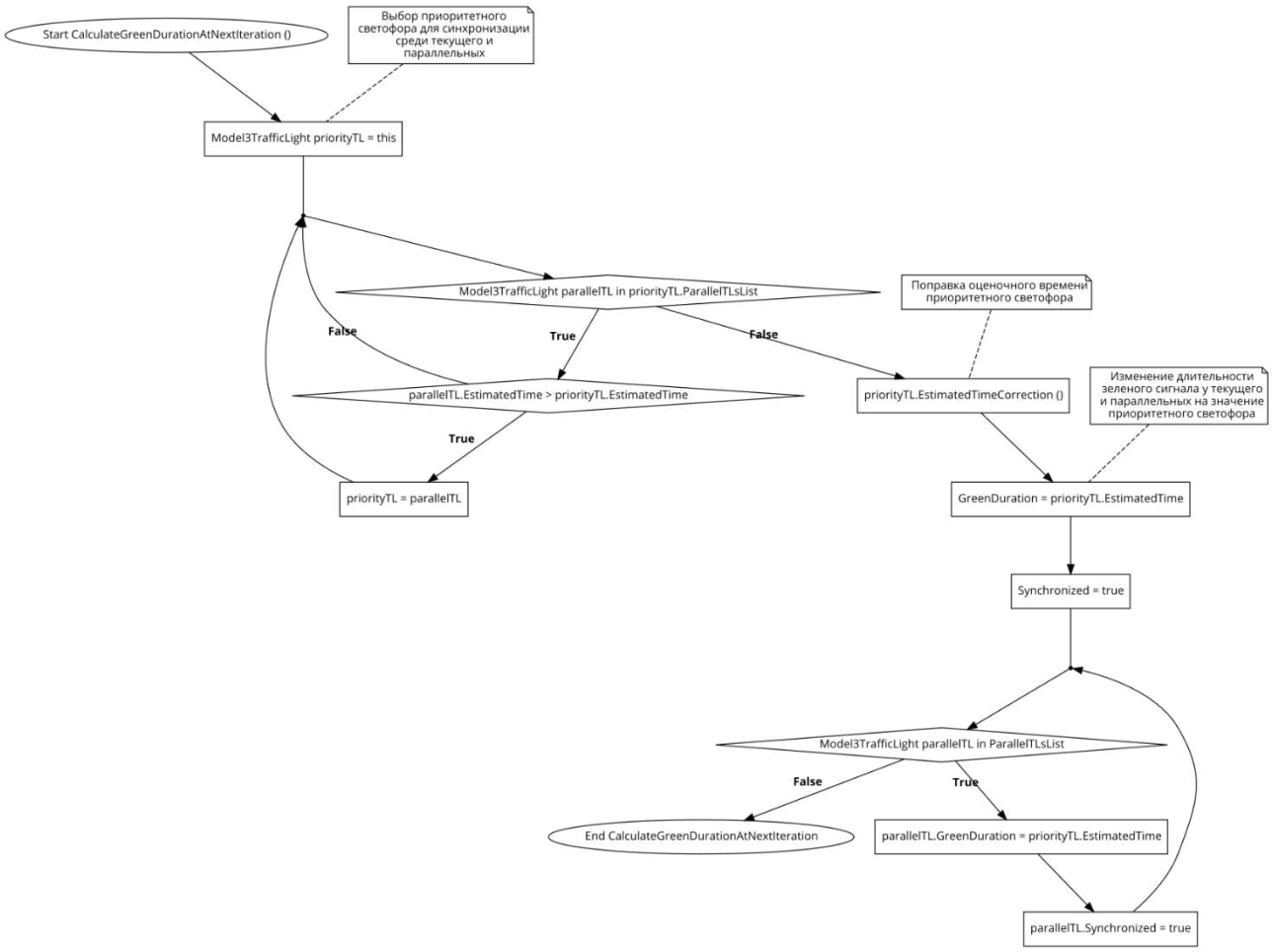

Рисунок 22, лист 4

4.4 Тестирование программного продукта

Цель: тестирование проводилось с целью облегчить разработку приложения путем контроля ключевых модулей, работа которых может быть нарушена при внесении изменений в код.

Вид тестирования: модульное тестирование (unit testing).

Назначение тестовых сценариев: проверка правильности выходных данных с ожидаемым результатом у определенных методов.

Организация работ, основной график выполнения: данный пункт включает в себя названия тестируемых модулей, сценарии, входные и выходные данные (таблица 4, таблица 5), а также общий план выполняемых тестов (рисунок 23).

Таблица 4 – Тестирование компонента Route.cs

Название тестируемого компонента		Входные данные компонента	
Название тестируемого модуля	Назначение тестового сценария	Входные данные модуля	Ожидаемые выходные данные модуля
Route.cs		a "0 0 - 0 0,5"; b "0 0 – 0,5 0"; c "0 0 - 0,5 0,5"	
FindByPoints	Тестирование поиска маршрута в списке маршрутов по двум концевым точкам	0 0 - 0 0,5	a
		0 0 - 0,5 0	b
		0 0 - 0,5 0,5	c
IsContainsPoint	Тестирование проверки на наличие концевой точки в списке существующих маршрутов	0 0	true
		0 1	false
FindStartRoutes	Тестирование поиска стартовых отрезков (точек входа для авто)		a, c
FindCrossingRoutes	Тестирование поиска пересекающих маршрутов для заданного	r1 "10 10 - 20 10"; r2 "30 10 - 40 10"	
		r1	
		r1 "10 10 - 20 10"; r2 "15 20 - 20 20"	
		r1	
		r1 "10 10 - 20 10"; r2 "15 10 - 30 10"	
		r1	r2
		r1 "10 10 - 20 10"; r2 "30 10 - 40 10"	
		r1	r2
		r1 "10 10 - 20 10"; r2 "20 10 - 30 10"	
		r1	r2
		r1 "10 10 - 20 10"; r2 "10 10 - 20 10"	
		r1	
		r1 "10 10 - 10 20"; r2 "10 30 - 10 40"	
		r1	
		r1 "10 10 - 10 20"; r2 "20 15 - 20 30"	
		r1	r2
		r1 "10 10 - 10 20"; r2 "10 15 - 10 30"	
		r1	r2
		r1 "10 10 - 10 20"; r2 "10 20 - 10 30"	
		r1	r2
		r1 "10 10 - 10 20"; r2 "10 10 - 10 20"	
		r1	r2
		r1 "10 10 - 20 10"; r2 "11 15 - 5 5"	
		r1	
		r1 "10 10 - 20 10"; r2 "30 5 - 40 15"	
		r1	
		r1 "10 10 - 20 10"; r2 "15 15 - 10 10"	
		r1	r2
		r1 "10 10 - 20 20"; r2 "30 30 - 60 10"	
		r1	
		r1 "10 10 - 50 15"; r2 "30 30 - 20 20"	
		r1	r2
		r1 "50 50 - 30 30"; r2 "30 30 - 20 20"	
		r1	r2
		r1 "30 30 - 50 50"; r2 "40 40 - 20 20"	
		r1	r2
		r1 "40 40 - 50 40"; r2 "40 30 - 40 50"	
		r1	r2
		r1 "40 40 - 50 40"; r2 "50 40 - 60 60"	
		r1	
		r1 "40 40 - 50 40"; r2 "39 30 - 39 50"	
		r1	

Таблица 5 – Тестирование компонента TrafficLight.cs

Название тестируемого компонента		Входные данные компонента	
Название тестируемого модуля	Назначение тестового сценария	Входные данные модуля	Ожидаемые выходные данные модуля
TrafficLight.cs		TL1 "X=100,5; Y=68; Speed=8; Type=up; GD=25"; TL2 "X=100,5; Y=71,5; Speed=8; Type=usual; GD=25"; TL3 "X=107,5; Y=58; Speed=0; Type=usual; RD=25"; TL4 "X=110,5; Y=58; Speed=0; Type=right; RD=25"; TL5 "X=110,5; Y=78; Speed=0; Type=left; RD=25"; TL5 "X=113,5; Y=78; Speed=0; Type=usual; RD=25"; TL6 "X=113,5; Y=78; Speed=0; Type=usual; RD=25"; TL7 "X=120,5; Y=65; Speed=8; Type=usual; GD=25"; TL8 "X=120,5; Y=68; Speed=8; Type=down; GD=25"; R1 "64 68 - 106 68"; R2 "64 71,5 - 103 71,5"; R3 "103 65 - 64 65"; R4 "103 71,5 - 107,5 76"; R5 "103 71,5 - 118 71,5"; R6 "106 68 - 113,5 60,5"; R7 "107,5 22 - 107,5 60,5"; R8 "107,5 60,5 - 103 65"; R9 "107,5 60,5 - 107,5 76"; R10 "107,5 76 - 107,5 114,5"; R11 "110,5 22 - 110,5 63,5"; R12 "110,5 63,5 - 118 71,5"; R13 "110,5 72,5 - 103 65"; R14 "110,5 114,5 - 110,5 72,5"; R15 "113,5 60,5 - 113,5 22"; R16 "113,5 76 - 118 71,5"; R17 "113,5 76 - 113,5 60,5"; R18 "113,5 114,5 - 113,5 76"; R19 "115 68 - 107,5 76"; R20 "118 65 - 113,5 60,5"; R21 "118 65 - 103 65"; R22 "118 71,5 - 130,5 71,5"; R23 "130,5 65 - 118 65"; R24 "130,5 68 - 115 68";	
FindDependentTLs	Тестирование поиска зависимых светофоров	TL1.ParallelTLsList	TL2, TL7, TL8
		TL1.PerpendicularTLsList	TL3, TL4, TL5, TL6
		TL2.ParallelTLsList	TL1, TL7, TL8
		TL2.PerpendicularTLsList	TL3, TL4, TL5, TL6
		TL3.ParallelTLsList	TL4, TL5, TL6
		TL3.PerpendicularTLsList	TL1, TL2, TL7, TL8
		TL4.ParallelTLsList	TL3, TL5, TL6
		TL4.PerpendicularTLsList	TL1, TL2, TL7, TL8
		TL5.ParallelTLsList	TL3, TL4, TL6
		TL5.PerpendicularTLsList	TL1, TL2, TL7, TL8
		TL6.ParallelTLsList	TL3, TL4, TL5
		TL6.PerpendicularTLsList	TL1, TL2, TL7, TL8
		TL7.ParallelTLsList	TL1, TL2, TL8
		TL7.PerpendicularTLsList	TL3, TL4, TL5, TL6
		TL8.ParallelTLsList	TL1, TL2, TL7
		TL8.PerpendicularTLsList	TL3, TL4, TL5, TL6
FindInfluentialRoutes	Тестирование поиска влияющих маршрутов	TL1	R9, R12, R13, R17, R20, R21
		TL2	R9, R12, R13, R16, R17, R19
		TL3	R21, R13, R6, R5, R4, R19
		TL4	R21, R6, R19, R17, R5, R16
		TL5	R5, R19, R6, R9, R21, R8
		TL6	R5, R12, R19, R21, R6, R20
		TL7	R17, R6, R12, R9, R13, R8
		TL8	R17, R12, R13, R5, R9, R4
SignalTiming	Тестирование синхронизации сигналов текущего и зависимых светофоров при изменении длительности сигнала	GD TL1=10	10, 25, 25, 25, 25, 25, 25, 25
		GD TL1=10; TL2=15	10, 15, 25, 25, 25, 25, 25, 25
		GD TL1=10; TL2=15; TL7=10	10, 15, 25, 25, 25, 25, 10, 25
		GD TL1=10; TL2=15; TL7=10; TL8=5	10, 15, 15, 15, 15, 15, 10, 5
		RD TL3=10; TL4=15; TL5=5; TL6=10	5, 5, 10, 15, 5, 10, 5, 5
		GD TL1=5; TL2=5; TL7=15; TL8=20	5, 5, 20, 20, 20, 20, 15, 20
		GD TL8=10	5, 5, 15, 15, 15, 15, 15, 10
		RD TL3=50; TL4=60; TL5=70;TL6=70	50, 50, 50, 60, 70, 70, 50, 50
		RD TL3=80	60, 60, 80, 60, 70, 70, 60, 60

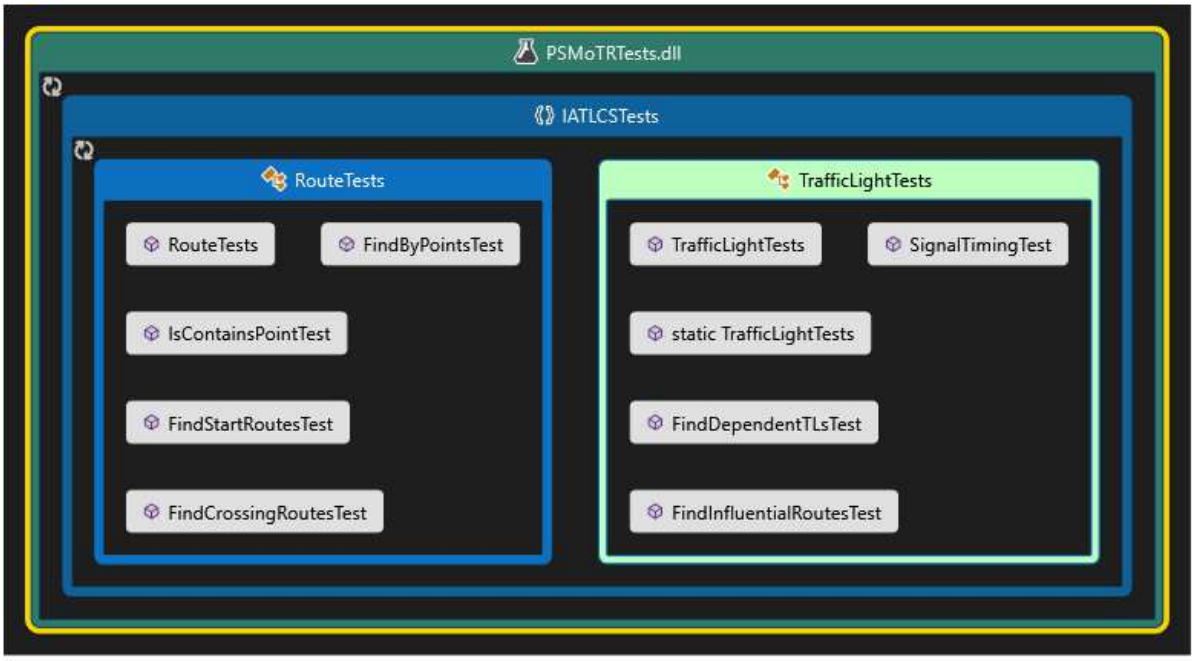

Рисунок 23 – Карта тестов проекта “PSMoTRTests” в Visual Studio

Инструкции по тестированию: наименование тестовых модулей/компонентов должно совпадать с наименованием тестируемого объекта, а также иметь приписку “Test”. Данные каждого тестового сценария инициализируются либо глобально в тестируемом компоненте, либо локально внутри тестового модуля. Для запуска тестирования достаточно выбрать «Выполнить все» или выбрать определенный тест во вкладке тестирования в проекте.

Используемые технологические средства и методики тестирования: тестирование производилось в проекте модульного тестирования “PSMoTRTests” в общем решении с основным проектом при помощи встроенного фреймворка Microsoft Test Framework. Правила

Детализация процесса тестирования: тестирование можно запускать вручную и ознакомиться с результатами в специальном окне, где также можно и узнать, где возникли проблемы с помощью добавленной трассировки в тестах (рисунок 24).

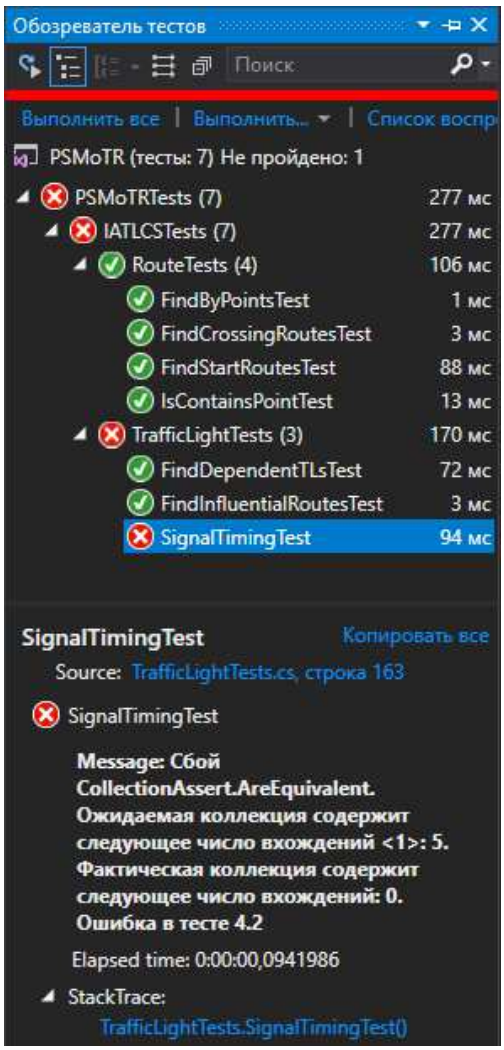

Рисунок 24 – Пример тестирования

4.5 Проектирование интерфейсов программного продукта

Интерфейсы проектировались согласно пункту 2.4 текущего документа.

Форма “MenuForm” представляет из себя окно, в котором есть «Панель управления», через которое выбирается и запускается/останавливается модель, а также настраиваются необходимые параметры генерации автомобилей и отображается общий таймер работы модели. В разделе «Управления моделью» пользователь выбирает, что отображать или скрыть на самой модели. Также на ней пользователь может перейти к редакторам маршрутов и светофоров.

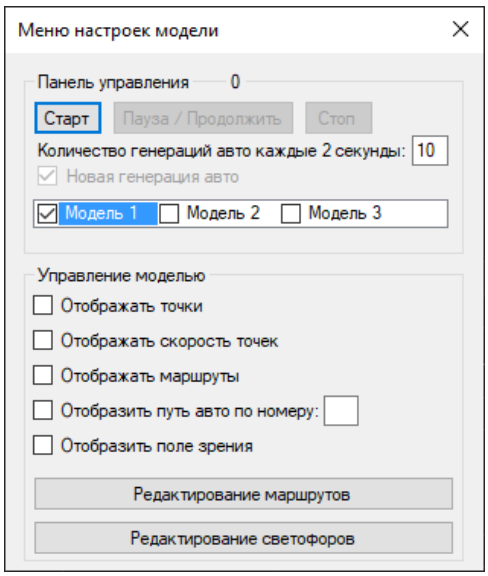

Рисунок 25 – Интерфейс “MenuForm”

Форма “ModelForm” представляет из себя окно, где происходит визуализация моделируемого процесса (рисунок 26). На нем используется фоновое изображение схемы дорог и отображение элементов по умолчанию и тех, что выбрал пользователь.

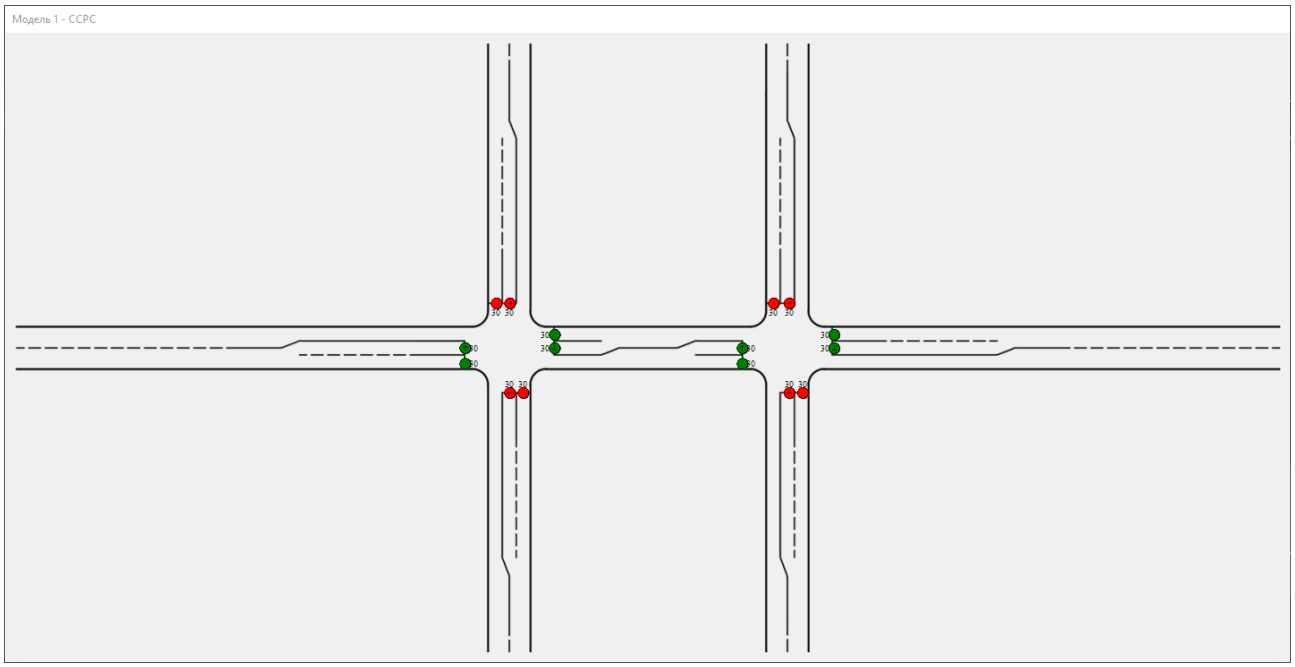

Рисунок 26 – Интерфейс “ModelForm”

Форма “RoutesEditForm” представляет из себя окно, где отображается список всех существующих маршрутов в виде таблицы, которые пользователь может изменять, добавлять и удалять, а также сохранять (рисунок 27).

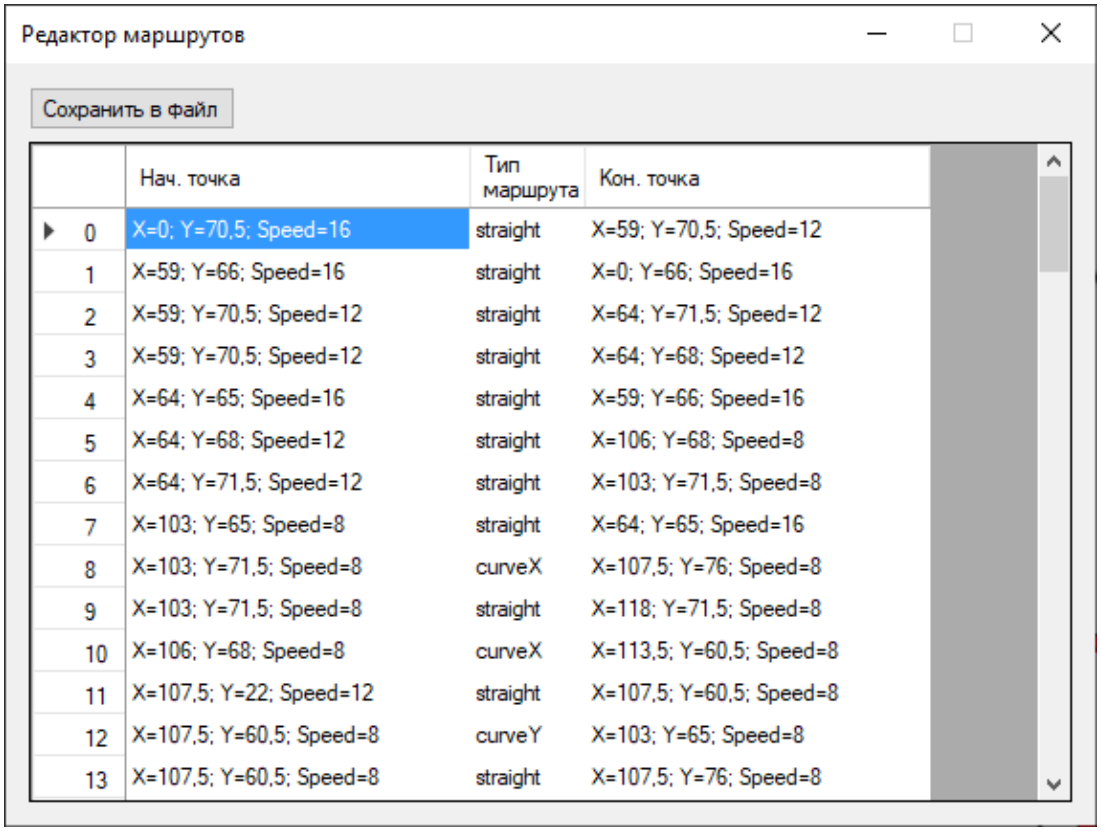

Рисунок 27 – Интерфейс “RoutesEditForm”

Форма “TrafficLightsEditForm” представляет из себя схожее во многом с “RoutesEditForm” окно, где отображается список всех существующих светофоров в виде таблицы, которые пользователь может изменять, добавлять и удалять, а также сохранять (рисунок 28).

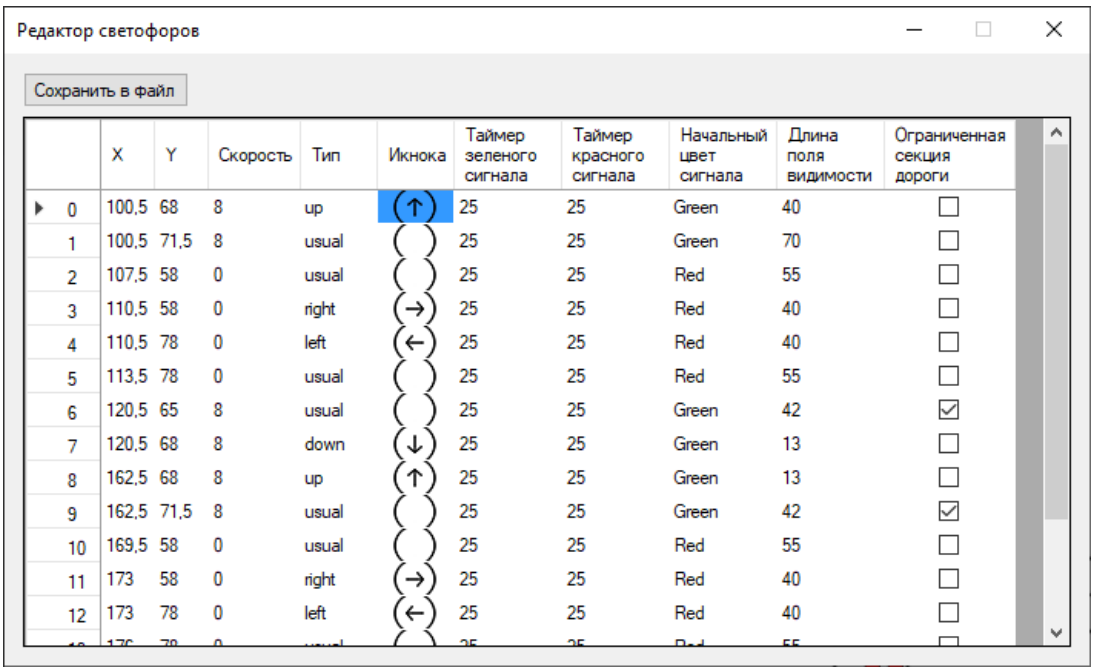

Рисунок 28 – Интерфейс “TrafficLightsEditForm”

4.6 Описание информационно-логической модели базы данных

В программном решении база данных представляет из себя локальное хранилище данных в текстовом файлах вместо конкретной базы данных. Такой выбор осуществлен в пользу того, что это дает возможность хранить данные без установки дополнительной системы управления базой данных. Также программная модель имеет незначительное количество данных, которые не требуют дополнительной защиты и занимают малый объем памяти, и такой способ хранения увеличивает скорость процесса работы с данными. База данных же может занимать объем памяти больше, чем занимают сами данные.

Работа с файлами представляет из себя процесс считывания и записи данных (рисунок 29). Каждый объект в файле представлен в виде строки, где каждое свойство отделено специальным разделителем. При считывании объектов они преобразуются в объекты класса при помощи специального парсера и хранятся в списке на протяжении сеанса работы программы. При записи же происходит обратный процесс преобразования объектов класса в их строковый формат.

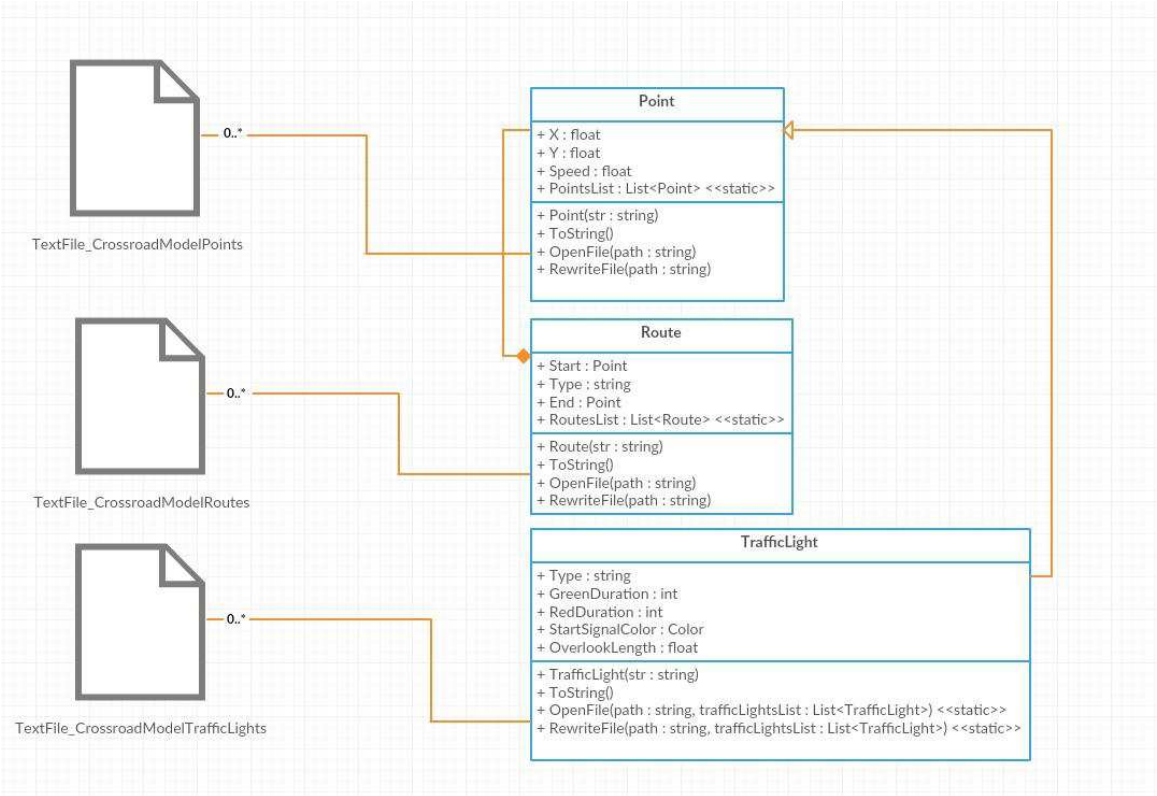

Рисунок 29 – Схема структуры работы с данными

5 Описание разработанного программного продукта

Область применения:

Разработанный программный продукт ИМПРТД предназначен для разработки и исследования различных алгоритмов регулирования светофоров, в который входят следующие процессы:

- моделирование транспортного движения;
- создание маршрутов движения транспорта;
- размещение и регулирование светофоров.

Краткое описание возможностей:

Разработанная программа предлагает решение проблемы внедрения светофоров, являющихся частью ИТС в виде программной модели ИМПРТД.

Данная модель позволяет проводить эксперименты с различными светофорами, которые работают по разработанным пользователем алгоритмам, с целью облегчить их внедрение в реальный процесс. В ней происходит моделирование движения транспорта, а также регулируется интенсивность потока, что создает вполне схожую с реальностью ситуацию на дороге, с которой и должны справляться разработанные светофоры с их инновационными алгоритмами.

Для внедрения данной системы, необходим лишь один пользователь с целью поддерживать работоспособность системы. Для работы с программным продуктом ему должны быть доступны следующие функции: управление моделью, редактирование маршрутов, редактирование светофоров, а также задание загруженности машинами.

Уровень подготовки пользователя:

В состав персонала, необходимого для обеспечения эксплуатации ИМПРТД, необходимо выделить только одного опытного пользователя.

Данное лицо должно выполнять следующие функциональные обязанности:

- настраивать и управлять моделью;

- изменять алгоритмы светофоров в исходном коде программы;

- редактировать маршруты по мере необходимости;

- редактировать светофоры по мере необходимости.

В состав требований к квалификации, предъявляемых к пользователю, включаются следующие основные требования:

- умение работать с компьютером под управлением ОС Windows;

- знание правил дорожного движения;

- умение работать в среде Visual Studio;

- навыки программирования на языке C#.

Перечень эксплуатационной документации, с которой необходимо ознакомиться пользователю:

Файл “readme.md” в репозитории программного продукта.

Виды деятельности и функции, для автоматизации которых предназначена программа:

Данная система предназначена для автоматизации следующих процессов:

- управления настройками модели;

- отображения окна с визуализацией моделируемого процесса;

- настройки маршрутов для движения транспорта;

- настройки параметров светофора;

- переключения алгоритмов регулирования светофоров;

- имитации движения транспорта и работы светофоров.

Условия, при которых обеспечивается применение программы:

Программные требования к системе:

- ОС Windows, начиная от версии XP и выше;

- среда разработки Visual Studio;

- .Net Framework, начиная от версии 4.7.2 и выше.

Аппаратные требования к системе:

Предъявляются минимальные системные требования, соответствующие используемой ОС, а также:

- дополнительно 128 МБ ОЗУ к минимальным требованиям ОС;

- 50 МБ свободного пространства на жестком диске.

Состав и содержание дистрибутивного носителя данных:

В состав дистрибутива (репозитория проекта) входит проект Visual Studio с исходным кодом программы.

Порядок загрузки данных и программ:

Для того, чтобы запустить ИМПРТД, необходимо открыть проект “PSMoTR.sln” в Visual Studio и выполнить запуск программы или же в корневой папке проекта найти и открыть папку “/bin/debug” / “/bin/release” и запустить файл “PSMoTR.exe”.

Порядок проверки работоспособности:

Программное обеспечение работоспособно, если в результате действий пользователя, изложенных в пункте «Порядок загрузки данных и программ», на экране монитора отобразились окна приложения без выдачи пользователю сообщений о сбое в работе.

Описание всех выполняемых функций (таблица 6):

Таблица 6 – Функции программного продукта

Функции	Описание
Настройка модели	Выбор количества генерируемых автомобилей, выбор модели, выбор отображаемых элементов, открытие окон редактирования маршрутов и светофоров
Моделирование	Запуск моделирования, приостановка / возобновление моделирования, остановка моделирования
Редактирование маршрутов / светофоров	Добавление, изменение и удаление соответствующих объектов
Изменение алгоритмов регулирования светофоров	Изменение алгоритма работы светофора выбранной модели в файле класса с исходным кодом

Описание операций технологического процесса обработки данных, необходимых для выполнения функций (таблица 7):

Таблица 7 – Операции функций программного продукта

Функция операций				
Наименование	Условия выполнения	Подготовительные действия	Основные действия	Заключительные действия
Настройка модели				
Выбор количества генераций авто каждые 2 секунды	Выбрана опция «Новая генерация авто», модель не запущена	Выбрать поле для ввода напротив опции	Ввести число (от 1 до 30)	Убрать фокус с поля для ввода
Выбор модели	Модель не запущена	-	Выбор одной из опций выбора модели	-
Отображение элементов на модели	-	-	Выбор опций для отображения	-
Открытие редактора маршрутов / светофоров	Модель не запущена	-	Нажать на кнопку «Открыть редактор маршрутов» / «Открыть редактор светофоров»	Закрыть окно редактирования по завершению действий там
Моделирование				
Запуск моделирования	Модель не запущена	-	Нажатие на кнопку «Старт»	-
Приостановка / возобновление моделирования	Модель запущена	-	Нажатие на кнопку «Пауза / Продолжить»	-
Остановка моделирования	Модель запущена	-	Нажатие на кнопку «Стоп»	-
Редактирование маршрутов / светофоров				
Добавление маршрута / светофора	Открыт редактор маршрутов / светофоров	Выбор последней строки в таблице	Ввод параметров в строку	Нажать на клавишу “Enter”
Изменение маршрута / светофора	Открыт редактор маршрутов / светофоров	Выбор необходимой строки в таблице	Изменить параметры строки	Нажать на клавишу “Enter”
Удаление маршрута / светофора	Открыт редактор маршрутов / светофоров	Выбор необходимой строки в таблице	Нажать клавишу “Delete”	-
Сохранение маршрутов / светофоров	Открыт редактор маршрутов / светофоров	-	Нажать на кнопку «Сохранить в файл»	Нажать на кнопку «ОК» в окне уведомления об успешном сохранении
Изменение алгоритмов регулирования светофоров				
Редактирование алгоритма светофора модели	Программа не запущена, открыт редактор исходного кода	Выбор и открытие исходного кода класса модели светофора	Внести изменения в алгоритм	Сохранить изменения

Аварийные ситуации:

При сбое в работе аппаратуры восстановление нормальной работы системы должно производиться после:

- перезагрузки операционной системы;

- запуска исполняемого файла системы.

При ошибках в работе аппаратных средств (кроме носителей данных и программ) восстановление функции системы возлагается на ОС.

При ошибках, связанных с программным обеспечением (ОС и драйверы устройств), восстановление работоспособности возлагается на ОС.

При неверных действиях пользователей, неверных форматах или недопустимых значениях входных данных, система выдает пользователю соответствующие сообщения, после чего возвращается в рабочее состояние, предшествовавшее неверной (недопустимой) команде или некорректному вводу данных. В случае иной неизвестной ошибки следует сообщить разработчику программы о проблеме.

Рекомендации по освоению:

Для успешного освоения программы ИМПРТД необходимо изучить следующее:

- правила дорожного движения Российской Федерации;
- руководство по обращению и работе с ОС Windows;
- программирование на языке C#;
- настоящий раздел «Описание разработанного программного продукта».

6 Исследование алгоритмов регулирования светофоров

Для исследования разработанного алгоритма ВАСУРС с помощью разработанной программной модели был проведен ряд экспериментов, заключающихся в симуляции одной и той же ситуации на заданном участке дороги. В качестве сравнительных алгоритмов регулирования светофоров для ВАСУРС выбраны уже упомянутые ранее ССУРС и АСУРС.

Входными данными для эксперимента являются задаваемое количество генерации автомобилей каждые 2 секунды (интенсивность потока) со случайными маршрутами. В качестве участка дороги для моделирования эксперимента выбраны два симметричных схематических перекрестка, между которыми ограниченный по расстоянию участок дороги. С каждой стороны перекрестка трехполосная дорога, где две сонаправленные полосы регулируются светофорами вида «стрелка налево» и «круглый светофор» (пункт 6.1 ПДД РФ [22]).

Выходными данными эксперимента являются:

- время, за которое модель разрешила дорожную ситуацию: эффективнее будет тот алгоритм, с которым модель затратит меньшее время до момента, когда все автомобили закончат свое движение и дорога окажется пустой;

- количество заторов на перекрестке: случай, когда автомобили не могут поместиться на ограниченном длиной участке дороги и вынуждены оставаться на перекрестке (рисунок 30), что, примечательно, является нарушением пункта 13.2 ПДД РФ, однако в жизни зачастую возникает и приводит к заторам [23].

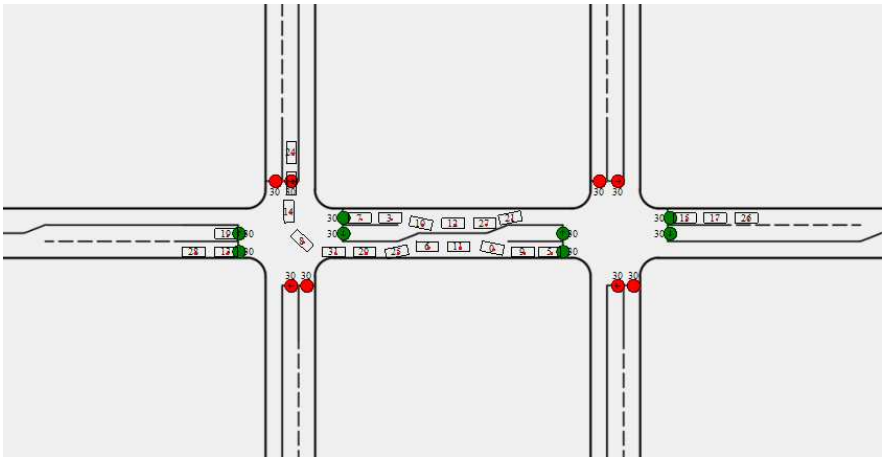

Рисунок 30 – Пример затора на перекрестке

Также возможна ситуация, когда движение прекращается совсем, когда встречаются четыре машины с различных направлений из-за затора на перекрестке, где каждый уже не имеет возможности уступить другому, действуя согласно алгоритму движения автомобилей программной модели (рисунок 31). В этом случае нет смысла измерять время и это отражает неэффективность алгоритма регулирования светофора, если возникает данная ситуация.

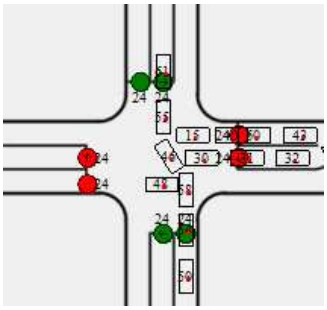

Рисунок 31 – Пример полного прекращения движения

Было проведено 12 экспериментов. Таблица 8 отображает результаты исследования. Если возникала ситуация, приводящая к полному прекращению движения (рисунок 31), то в колонке «Время» ставился «-» и в скобках указывалось время, когда это произошло.

Таблица 8 – Результаты исследования моделей ССУРС, АСУРС и ВАСУРС

№	Интенсивность потока	ССУРС		АСУРС		ВАСУРС	
		Время (с)	Кол-во заторов	Время (с)	Кол-во заторов	Время (с)	Кол-во заторов
1	5	87	0	88	0	87	0
2	5	90	0	75	0	75	0
3	5	97	0	97	0	97	0
4	10	150	1	120	0	120	0
5	10	145	0	119	0	128	0
6	10	145	0	130	0	125	0
7	15	145	0	125	0	143	0
8	15	266	6	187	2	188	1
9	15	161	1	148	0	155	0
10	20	- (144)	1	- (133)	3	228	0
11	20	- (150)	1	188	2	213	0
12	20	- (112)	1	- (120)	1	178	0

В экспериментах 1-3 при низкой интенсивности движения каждый из алгоритмов успешно справился с выполнением своей задачи. По мере роста интенсивности движения в экспериментах 2, 4-6, 8, 9 АСУРС и ВАСУРС показали результат лучше, чем ССУРС. В 7 эксперименте ВАСУРС показал результаты чуть хуже, чем АСУРС, но это скорее частный случай. Однако по результатам экспериментов 10, 12 при дальнейшем росте интенсивности ССУРС перестал эффективно решать дорожную ситуацию, при этом каждый раз прекращалось движение (рисунок 31), причем АСУРС также допускал возникновение такой ситуации. Однако тут ВАСУРС показал наилучшие результаты, так как он не допускал возникновения затора на перекрестке, и, следовательно, прекращения движения, но если АСУРС не сталкивается с вышеописанными ситуациями, то ВАСУРС теряет преимущество перед ним, как это видно в эксперименте 11.

ЗАКЛЮЧЕНИЕ

Результатом выполнения данной работы стал разработанный программный продукт ИМПРТД. Для достижения первой поставленной цели, поставленной в рамках выполнения выпускной квалификационной работы, были выполнены следующие задачи:

- проанализирована предметная область и сделано заключение о необходимости разработки собственной программы для моделирования;
- определены спецификации требований разработанного программного продукта;
- определены обеспечивающие элементы программного объекта проектирования: язык и среда программирования, система управления проектом;
- выполнена реализация программного продукта в соответствии с процессами проектирования;
- описан разработанный программный продукт.

Также была достигнута поставленная цель по программной реализации алгоритмов ССУРС, АСУРС и ВАСУРС в разработанном программе ИМПРТД.

Для достижения последней цели по исследованию и сравнению разработанных алгоритмов был проведен ряд экспериментов, по которым сделаны следующие выводы: установлено, что если интенсивность потока низкая, то ССУРС является приемлемым алгоритмом и может эффективно справляться со своей задачей. По мере роста интенсивности движения целесообразнее перейти на АСУРС, поскольку он быстрее и избирательнее разгружает транспортный поток, а сигналы светофоров не горят «вхолостую», как это наблюдается в ССУРС. Однако при очень высокой интенсивности потока АСУРС может допускать заторы на перекрестках, если имеются ограниченные сегменты дорог, что может привести к трудноразрешимым ситуациям. Но с такой проблемой может справиться авторский алгоритм ВАСУРС. Исходя из этого, имеет смысл внедрить в реальный процесс ВАСУРС, поскольку он доказал

свою эффективность работы с высокой интенсивностью потока на перекрестках с ограниченным расстоянием между ними.

Полученный в ходе разработки программный продукт может быть использован для разработки и исследования различных алгоритмов регулирования светофоров, а описанный алгоритм ВАСУРС может быть использован на реальных системах регулирования светофоров.

Также планируется в ближайшем будущем провести дальнейшие улучшения возможностей редакторов маршрутов и светофоров, добавить возможность добавления собственных изображений, по которым будет строиться модель. Есть планы по улучшению процесса имитации транспортного движения, в который входит улучшение существующего алгоритма движения и увеличение количества объектов моделирования. Эффективным решением в будущем будет подключение нейросетей для классификации транспорта и анализа времени, за которое требуется транспорту для преодоления регулируемого перекрестка, что улучшит вычисление времени длительности сигнала для адаптивных светофоров. Рассматривается вариант развития идеи и принятия участия в реализации взаимодействия между светофорами в известных программных решениях моделирования, используя наработки программы ИМПРТД.

СПИСОК СОКРАЩЕНИЙ

HID: устройство для взаимодействия с человеком (англ. Human Interface Device).

IEEE: институт инженеров электротехники и электроники (англ. Institute of Electrical and Electronics Engineers).

LINQ: язык интегрированных запросов (англ. Language-Integrated Query).

SRS: спецификация требований программного обеспечения (англ. Software Requirements Specification).

UI: пользовательский интерфейс (англ. User Interface).

XML: расширяемый язык разметки (англ. eXtensible Markup Language).

АСУРС: адаптивная система управления работы светофоров.

ВАСУРС: взаимодействующая адаптивная система управления работы светофоров.

ВКР: выпускная квалификационная работа.

ИМПРТД: имитационная модель процесса регулирования транспортного движения.

ИТС: интеллектуальная транспортная система.

МБ: мегабайт.

ОЗУ: оперативное запоминающее устройство.

ОС: операционная система.

ПО: программное обеспечение.

ССУРС: стандартная система управления работы светофоров.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Владимиров С.Н. Транспортные заторы в условиях мегаполиса // Известия МГТУ "МАМИ", Т. 3, № 1(19), 2014. С. 77-84.

2. Евсеева А.А., Казаков А.А. Методы решения проблемы автомобильных пробок в Саратове на примере мирового опыта // Научно-методический электронный журнал "Концепт", Т. 20, 2014. С. 3531-3535.

3. Комаров Ю.Я., Алшахван А. Изучение влияния светофоров на ситуацию дорожного движения на перекрестке у Российского консульства в г. Алеппо (САР) // Молодой ученый, № 18, 2018. С. 71-75.

4. Основные принципы работы светофора [Электронный ресурс] // Дор Технологии: [сайт]. [2017]. URL: https://www.dortec.ru/statji/osnovnye-printsipy-raboty-svetofora.html (дата обращения: 28.05.2019).

5. Светофор: функции, виды, регулирование [Электронный ресурс] // ТрансСпот: [сайт]. [2015]. URL: http://transspot.ru/2015/01/11/svetofor-funkcii-vidy-regulirovanie/ (дата обращения: 28.05.2019).

6. Солоницына К.А. "Умный светофор" как часть интеллектуальной транспортной системы // Студенческий форум: электрон. научн. журн., № 8(29), 2018.

7. Прошкина Е.Н., Балашова И.Ю. Исследование моделей обнаружения закономерностей в потоке движущихся объектов // Научный журнал, Т. 6, № 2, 2018. С. 198-207.

8. Приступа А.В. Имитационная модель перекрестка с двухфазным светофорным регулированием // Вестник Томского государственного университета, № 3(24), 2013. С. 139-142.

9. Пилецкая А.А. Разработка алгоритма управления системой светофоров // GitHub. 2010. URL: http://aurusov.github.io/diploma/2010/Разработка%20алгоритма%20управления%20системой%20светофоров%20-

%20Пилецкая%20Анастасия%20-%20Шаг/Записка.pdf (дата обращения: 28.05.2019).

10. Модель светофора [Электронный ресурс] // Микроконтроллеры: [сайт]. [2013]. URL: https://mcucpu.ru/index.php/project2/ustrojstva-indikatsii/132-traffic-light (дата обращения: 28.05.2019).

11. Котов А.Н. Моделирование дорожного движения на многополосной магистрали при помощи двумерного вероятностного клеточного автомата с тремя состояниями, Университет ИТМО. Кафедра "Технологии программирования", Санкт-Петербург, Магистерская диссертация 2008.

12. Дорожное движение [Электронный ресурс] // AnyLogic: [сайт]. [2019]. URL: https://www.anylogic.ru/road-traffic/ (дата обращения: 29.05.2019).

13. Vistro [Электронный ресурс] // PTV Partner: [сайт]. [2019]. URL: http://ptv-vision.ru/produkty/vistro/ (дата обращения: 29.05.2019).

14. Aimsun Auto [Электронный ресурс] // Aimsun: [сайт]. [2019]. URL: https://www.aimsun.com/aimsun-auto/ (дата обращения: 29.05.2019).

15. IEEE 830-1998 - IEEE Recommended Practice for Software Requirements Specifications [Электронный ресурс] // IEEE Standards Association: [сайт]. [2019]. URL: https://standards.ieee.org/standard/830-1998.html (дата обращения: 30.05.2019).

16. Microsoft Visual Studio [Электронный ресурс] // Wikipedia: [сайт]. [2016]. URL: https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio (дата обращения: 31.05.2019).

17. Visual Studio [Электронный ресурс] // Microsoft: [сайт]. [2019]. URL: https://visualstudio.microsoft.com/ru/vs/ (дата обращения: 31.05.2019).

18. C Sharp [Электронный ресурс] // Wikipedia: [сайт]. [2018]. URL: https://ru.wikipedia.org/wiki/C_Sharp (дата обращения: 31.05.2019).

19. Краткий обзор языка C# [Электронный ресурс] // Microsoft Docs: [сайт]. [2019]. URL: https://docs.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/index (дата обращения: 31.05.2019).

20. Azure DevOps [Электронный ресурс] // Microsoft Azure: [сайт]. [2019]. URL: https://azure.microsoft.com/ru-ru/services/devops/?cdn=disable (дата обращения: 04.06.2019).

21. Обзор технологии DevOps [Электронный ресурс] // Microsoft Azure: [сайт]. [2019]. URL: https://azure.microsoft.com/ru-ru/overview/devops/?cdn=disable (дата обращения: 04.06.2019).

22. ПДД РФ, 6. Сигналы светофора и регулировщика [Электронный ресурс] // Официальный сайт компании «КонсультантПлюс»: [сайт]. [2019]. URL: http://www.consultant.ru/document/cons_doc_LAW_2709/4b7a10a56ed37080fc96999db5f3db6f3aa58cc6/ (дата обращения: 25.05.2019).

23. ПДД РФ, 13. Проезд перекрестков [Электронный ресурс] // Официальный сайт компании «КонсультантПлюс»: [сайт]. [2019]. URL: http://www.consultant.ru/document/cons_doc_LAW_2709/74cbe820904f4f8ce76047ddbd81d14c8b953d3e/ (дата обращения: 25.05.2019).

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт
Кафедра «Информатика»
кафедра

УТВЕРЖДАЮ
Заведующий кафедрой
_____ А.С. Кузнецов
подпись инициалы, фамилия
«05» 07 2019 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.04 «Программная инженерия»
код и наименование специальности
Программная модель имитации процесса регулирования транспортного
тема
движения с применением взаимодействующей адаптивной системы управления
работы светофоров

Руководитель _____ 05.07.19 старший преподаватель А.С. Михалев
подпись, дата должность, ученая степень инициалы, фамилия

Выпускник _____ 05.07.2019 Т.В. Родионов
подпись, дата инициалы, фамилия

Консультант _____ доцент, к.т.н. А.С. Кузнецов
подпись, дата должность, ученая степень инициалы, фамилия

Нормоконтролер _____ 05.07.19 доцент, к.т.н. О.А. Антамошкин
подпись, дата инициалы, фамилия

Красноярск 2019